Computational Gambling

Konstantinos Katsiapis

Introduction

Gambling establishments work with the central dogma of Percentage Payout (PP). They give back only a percentage of what they get. For example they could return only 80% of the collective receipts but the returns are unevenly distributed between the players, so some of them end up with huge winnings while most end up with small loses.

We have at our disposal data of several Shops (mini casinos) from different areas and countries. Our goal is to apply machine learning algorithms on the data and try to infer the effect of PP on the shop's performance. We are interested in predicting performance indicators such as profit or actual percentage.

Available Data

At any given Day, a Shop is instructed to play at a specific PP. However there is no guarantee that the actual PP will match the instructed one. This is largely due to the random nature of the games (which is a desired property), and partly due to the fact that some games don't follow their instructions (which is not desired but a reality nonetheless). Randomness on the behavior of a game conditioned on its instructed PP is desired in order to prevent determinism and guessing on the part of the players for the behavior of the game in a single Day. However, assuming a Shop is instructed to play at a specific PP for a long period of time, the actual PP is expected to converge to the instructed one. Intuitively we can think of the set_percentage as a "nudge" to the right direction that the Shop should follow. In reality the PP almost always changes from Day to Day, and the changes might be abrupt, but the overall effect averaged out over several Days should smooth out. One aspect of the paper will be to predict the actual percentage payout based on the instructed percentage payout. This will give as a side effect an indication of the degree to which the games follow their instructions.

For any given Shop and Day we have the following data available:

- set_percentage [Controllable] (this is the instructed PP for the Day)
- revenue [Observed]
- profit [Observed]
- shop_percentage [Observed] (this is the empirical percentage obtained from profit and revenue measures)
- games_percentage [Observed] (this is the empirical percentage obtained from profit revenue and gift measures and denotes the actual percentage at which the games played)

We currently have at our disposal data for 50,000 Days obtained from 400 Shops, yielding an average of 125 Days/Shop (all numbers are approximate).

Approach

Our initial approach is to perform regressions using set_percentage as the feature and one of revenue, profit, shop_percentage and games_percentage as the response variable. The rational behind this follows naturally from the fact that we can control the set_percentage and hope to be able to tune the remaining based on the first.

Predicting the profit based on the set_percentage might sound trivial at first, but it's anything but: The problem is that a Shop's revenue does not remain fixed but rather depends on set_percentage in direct and indirect ways: Surely a player will prefer a Shop where her chances of winning are higher (reflected on average by a higher set_percentage). Consequently, a low set_percentage for a Shop might imply higher profit in the short run but most probably a loss of customers and a smaller revenue and profit in the long run. Our goal is to be able to predict the profit based on set_percentage and perhaps find the optimal value of the second for long term equilibriums.

A secondary approach will involve extending the feature set of Controllable features (set_percentage) with Observed historical features (eg yesterdays revenue, profit, games_percentage etc).

Averaging of set_percentage

All the available data are exact for any given Day. However as already mentioned, predictions of response variables for any given Day are expected to be bad, since the games' behavior appears random in small periods of time and small amounts of money played. Naturally we would like to extend our time and point frame by looking at aggregates over a week or month. Aggregating profits, revenues and shop_percentage and games_percentage is trivial, but aggregating set_percentage is not. The naive approach would be to take a simple average of set_percentage. The problem with this approach is that all Days are treated equal. Assume for example that of the 7 Days in a week, the first 6 had a very low set_percentage with a small amount of played money and small profits. Further assume the 7th day had a high set_percentage with big amounts of played money and profits. A normal average of set_percentage will give the incorrect training example of low set_percentage yielding a high profit.

To avoid the problem we perform a weighted average of set_percentage based on the amount of money played under that set_percentage. Intuitively, when a Shop is instructed for a long period of time/money to play at a specific set_percentage it is expected to converge to that faster compared to small periods of time/money.

To avoid pathological situations where no amount of money was played in a given week we apply Laplace Smoothing in our averaging. A formula for averaging set_percentage is thus given by:

set_percentage := SUM((day_weight + 1) * set_percentage) / (SUM(day_weights) + COUNT(days))

This same technique is not only useful when averaging set_percentage across many Days, but also when averaging the value across multiple Shops (provided they share the same currency). We can thus apply our techniques to Shop groups (areas) or even entire countries by aggregating on a per currency-basis.

Models

Our models consist of Least Squares Fitting approaches. We limit ourselves to using Polynomial models of degree >= 2 and also Locally Weighted Linear Regression models (with Simple Unweighted Linear treated as a subcategory with equal weights).

We are using Ruby/GSL, an interface to the GNU Scientific Library, for our computations.

Polynomial models allow us to use a single feature while Locally Weighted Linear Regression models extend well to multiple features. Consequently, current Controllable features can easily be mixed with historic Controllable and Observed features (for more details look at Feature Selection section below).

Here are the exact choices of our models:

- 1 Linear Model
- 4 Polynomial Models (orders: {2 to 5, step1})
- 27 Locally Weighted Linear Models (taus: {0.1 to 0.9, step 0.1}, {1 to 9, step 1} and {10 to 90, step 10})

Model Selection

Standard methods for selecting the best model are employed. Specifically we use the following 5:

- 2 Simple Cross Validation Methods [SCV] (test data at 30% and 50%)
- 2 K-Fold Cross Validation Methods [KFCV] (k: {5, 10})
- 1 Leave One Out Cross Validation Method [LOOCV] (a special case of K-fold really)

In order to gather some intuition and knowledge on which is the best selection method for our data we are calculating the best model based on each of the selection methods and are recording its parameters, training and test error.

Feature Selection

Our initial approach is to use set_percentage as the only feature. We expand on that by adding historical values of set_percentage as additional features. For example when attempting to predict Day A's response variable we will use the set_percentage of Days A, A-1, A-2 etc. We evaluate each model using each of the selection methods on historical data with history values 1 to 8 (step 1). We do the same for weekly and monthly calculations.

Our secondary approach is to add Observed features into the historical information. For example we use the profit, revenue, shop_percentage and games_percentage of Days A, A-1, A-2 etc. Since the feature count is much larger now we only evaluate our models for history values 1 to 5 (step 1). Similarly for weekly and monthly calculations.

Note that there is no need to perform generalized Feature Selection techniques (like Forward Search, or Backward Search) as there is no combinatorial aspect (no combinatorial explosion) in the choices of set_percentage. The reason is that we expect set_percentages closer to the current time frame to be more important in affecting the response variable. There is thus no need to include a set_percentage of several time periods ago, without including all the set_percentages of the subsequent periods up until to the current time frame.

Further, historical profit and revenue are expected to have similar predictive power as they are linearly correlated (through the shop_percentage). Also a combination of shop_percentage and games_percentage is expected to have better predictive power than either of the two alone (as the combination conveys information about gifts).

Results

The astute reader (yes, that's all of you...) must have noticed that the number of experiments described in our approach is quite large. Indeed the number of fits we investigate is close to 200,000. Contrast that to the original number of Days for which we have data available. The vast majority of the experiments however are marginally useful and quite a few useless. They are there for completeness and also have an educational purpose in giving us general insight in bias, variance, feature and model selection. Further, quite a few Shops have a very small amount of data and the fits were largely useless.

Model Selection Method

Analysis of the experiments' data narrows down the comparison of feature selection methods down to three contenders:

- SCV (30%): Fast, Not-Robust
- KFCV (5): Medium speed, Not-Robust
- LOOCV : Slow, Robust

A method is considered Robust if its train and test error calculations do not vary a lot due to chance (random effects), even for a small number or training and test examples.

For the presentation of results, we decided to use LOOCV which although very slow for large data sets is always Robust due to its symmetric nature and so its train and test error can be trusted.

Time Frame

The hardest variable to predict was profit.

The best profit predictions at a Daily interval yield test errors of 50% or more.

The best profit predictions at a Weekly and Monthly interval yield test errors of 15% - 30%.

For the cases of profit and revenue predictions, we get better results when limiting our training and test set to the latest examples (eg last 20 months, last 40 weeks etc). This is explained by seasonality effects: Consider a Shop doing very well 3 years ago and poorly now. Using examples from the distant past will lead to biased optimistic predictions.

Predictions of shop_percentage and game_percentage don't suffer from this limitation (as they are relativistic in nature and magnitude agnostic).

Features

set_percentage alone has fairly good predictive power.

Adding history of set_percentage improves performance by 2 -3 percentage points in general. The best predictions usually involve a history of 1 to 3.

Adding history of Observed features such as shop_percentage and games_percentage further improves performance by another 1-2 percentage points in general. The best predictions usually involve a history of 1 to 2.

Surprisingly, inclusion of Observed features such as past revenue and profit rarely gives good results. Their inclusion causes severe overfitting, and test error increases by 5-10 percentage points. Attempts to prevent overfitting by penalizing high norm Thetas (a la MAP) failed; test error further increased.

Indicative Numerical Data

Analysis was done on a per Shop and on a per Currency level. We present below numerical results for Monthly and Weekly predictions of profit and games_percentage.

Graphical results are available in the Project Presentation at:

http://www.stanford.edu/~kkatsiap/CS229/project/CS229_Project_Presentation.pdf

Feature Acronyms: SP: set_percentage, SHP: shop_percentage, GP: games_percentage

Profit Prediction

Period	Group	Train error	Test error	Features	History	Limit	Model
by_month	Currency 114	15.0 %	17.1 %	SP	1	20	Linear
by_month	Currency 119	7.8 %	13.6 %	SP, SHP, GP	1	20	WLinear (0.2)
by_month	Shop 122	15.4 %	18.4 %	SP, SHP, GP	1	-	WLinear (0.3)
by_month	Shop 198	17.4 %	24.0 %	SP, SHP, GP	1	-	Linear
by_week	Currency 114	29.6 %	33.3 %	SP, SHP, GP	1	40	Linear
by_week	Currency 119	16.5 %	19 %	SP, SHP, GP	1	40	Linear
by_week	Shop 122	26.8 %	30.7 %	SP, SHP, GP	1	40	WLinear (0.3)
by_week	Shop 198	33.2 %	45.8 %	SP, SHP, GP	1	20	Linear

Games Percentage Prediction

Period	Group	Train error	Test error	Features	History	Limit	Model
by_month	Currency 114	3.5 %	4.0 %	SP	1	-	Linear
by_month	Currency 119	1.3 %	2.8 %	SP	1	-	WLinear (0.1)
by_month	Shop 122	3.2 %	4.1 %	SP	1	-	WLinear (0.2)
by_month	Shop 198	3.2 %	3.9 %	SP	1	-	WLinear (0.2)
by_week	Currency 114	6.9 %	7.1 %	SP	1	-	Linear
by_week	Currency 119	3.5 %	4.9 %	SP	1	-	WLinear (0.1)
by_week	Shop 122	8.3 %	8.8 %	SP	1	-	WLinear (0.2)
by_week	Shop 198	7.5 %	8.2 %	SP	1	-	WLinear (0.2)

Future Work

- **Comparison with Time Series Forecasting techniques:** We would expect our technique to perform better than raw Time Series Forecasting techniques as it takes into account set_percentage, a feature that has direct and indirect effect on the response variables, which Time Series Forecasting techniques do not.

Resources

Limited source code and documents are available at: http://www.stanford.edu/~kkatsiap/CS229/project/