‘Tennis Forecast’ - A Machine Learning Project
Full Project Report

Guilherme Klink (guilhermeklink2016@u.northwestern.edu)
Sherif Mostafa (sherifmostafa2015@u.northwestern.edu)
EECS-349-Machine Learning - Northwestern University

Project Overview

This project uses a machine learning approach to forecast Tennis match results (men and women) based on previous match statistics. Given two players and one of the 4 major Tennis tournaments (Australian Open, French Open - Roland Garros, Wimbledon and the US Open), the **provided application** (user manual attached in the appendix) will return the player that is more likely to win the match. In contrast to other sports - i.e. Baseball - publicly available statistical data for tennis is scarce, indicating that not much data analytics has been applied in the world of Tennis. The majority of Tennis forecasts, such as betting odds are obtained purely from player ranking, and number of wins/losses. The novelty of the approach taken in this project, is that it does not depend on the official ATP (Association of Tennis Professionals) or WTP (Women’s Tennis Association) player rankings or how many matches a player won or lost, but rather how they played. Instead, we look at more objective performance data (i.e. net points, unforced errors, serving percentages, serving direction, points won on the second serve...) .

A fundamental component of this project is to determine which features should be passed-on to the learning algorithm that was used to train the forecaster. The initial data scraped from various online sources provided up to 1000 features. Using correlation techniques, the most relevant features are extracted. By virtue of its interpretability, a decision tree learning algorithm is the preferred choice for this project. The features used contained both nominal and categorical types.

The majority of the data was crowd-charted, meaning that it contained plenty of inconsistencies and required intense preprocessing. Once the formatting was consistent, data from multiple sources was cross-checked to reduce the number of errors. Subsequently, the correlation index of each attribute to the classifier is determined. The attributes that exceed a certain correlation index threshold were selected and reexamined using the author's expertise in Tennis to avoid post hoc fallacies (also know as *post hoc ergo propter hoc*). The learned hypothesis was tested on the major ongoing tournament during the development of this project (2015 French Open), upon which the forecaster’s success is measured.

When it was tested on the 2015 Roland Garros tournament, the forecaster demonstrated a prediction accuracy of 89%.

A more thorough description of the building blocks that compose the project is provided in the next section.

‘Tennis Forecast’ - A Machine Learning Project
Full Project Report

Guilherme Klink (guilhermeklink2016@u.northwestern.edu)
Sherif Mostafa (sherifmostafa2015@u.northwestern.edu)
EECS-349-Machine Learning - Northwestern University

Project Details

Task description:

On a lower level the task description is as follows:

- Gather as much publicly available point-by-point Tennis data as possible (3032 matches producing over 200 points (instances) per match spreading over 140 features were gathered for this project).
- Preprocess the data to obtain consistent databases for each online resource.
- Crosscheck overlapping data between the sources to mitigate errors.
- Calculate the most significant features, that are likely to have a causal effect on the outcome of the match.
- Use a decision tree learning algorithm to obtain a human interpretable hypothesis that can be post analyzed and corrected if need be.
- Examine the decision tree’s accuracy in comparison to other learning algorithms.
- Test the forecaster’s accuracy on an ongoing major tournament.
- Develop an end-user application for the project

Applied software tools:

- SQLite3 scripts: Used to process, compile, and fix problems with the data.
- Python: Used to import .csv files into the databases and create the core of the prediction app.
- Weka: Used for its user friendly graphic interface and ease of use for examining a wide range of learning algorithms and getting a better sense of the data, as well as help us pick which attributes generate better predictors.
- Sci-kit-learn: As it is a python based machine learning tool, it is convenient to embed this learner in our python scripts and GUI (graphic user interface) to predict outcomes

Data Preprocessing:

Data sources:

- **“Tennis Match Charting Project”**:
  This is a crowd sourced data charting project that has very detailed point by point information for approximately 413 matches in the modern era. While it has a relatively small number of matches, the upshot is that each match provides highly detailed information (>100 features). The downside of this project, because it is crowd sourced, is that the data is extremely inconsistent and noisy. Filtering it was a pain.

‘Tennis Forecast’ - A Machine Learning Project
Full Project Report

Guilherme Klink (guilhermeklink2016@u.northwestern.edu)
Sherif Mostafa (sherifmostafa2015@u.northwestern.edu)
EECS-349-Machine Learning - Northwestern University

- **“Slam Point By Point Project”**:
  Hosted by the same group behind the Tennis Match Charting Project, this cleaner more consistent, but far less comprehensive data set was generated by scraping the official data provided at the end of each major tournament (from 2011-2015). Much of the most useful data isn't available for every tournament. “For instance, there is no first/second serve indicator for many events, and rally length is not included after the first few” - explain the hosts.
- **“ATP Ranking and Points Project”**:
  We **compiled this data set ourselves**, by looking up all the rankings and end of year points for each ATP player starting the year 2000 to date.
- **UC Irvine “Tennis Major Tournament Match Statistics Data Set”**
  The matches charted in this data set (the four major tournaments of 2013), are theoretically a subset of the “Slam Point By Point Project”. However, as it is part of a previous machine learning project, the data pre-filtered and can fill-in any missing data for that particular year in other datasets.
- **“tennis-data.co.uk” (a sports betting resource)**
  While this data has every match on the professional tour since 2000 charted, it contains only a handful of very primitive features about who won, lost, the score, and the betting odds.

Cleaning the data:

- For each of the aforementioned RAW Datasets, we have generated a database in SQLite3 (total of 140 tables). Contrary to most proprietary ODBC (Open Database Connectivity) tools, being an open source software, SQLite3 has no automated process for data import and processing. Since we are dealing with a large amount of data/tables, a python script was created to analyze each .csv file, create a table matching the shape of the data, and finally importing the data. SQL queries were used to compile the RAW tables into ‘views’. Additional python scripts were written to consistently format the data (different spelling of the same player; the same feature with different labels in the datasets; typos; blank inputs...) . The consistently formatted tables were cross-matched to mitigate the erroneous inputs (which is a major problem with crowd-sourced data.)

Feature selection:

- The scraped and cleaned data contains more than 150 features, of which the most relevant features are to be selected. Using a correlation formula,

$$r = \frac{n\sum xy - (\sum x)(\sum y)}{\sqrt{(\sum x^2) - (\sum x)^2}\sqrt{n(\sum y^2) - (\sum y)^2}}$$

Guilherme Klink (guilhermeklink2016@u.northwestern.edu)
Sherif Mostafa (sherifmostafa2015@u.northwestern.edu)
EECS-349-Machine Learning - Northwestern University

we obtain the correlation coefficient *r* of each feature *x* to the classifier *y*. Arbitrarily, features that demonstrate an absolute correlation index larger than a threshold $\theta=0.5$ are kept, while remaining features with a weaker correlation are discarded. The absolute value of *r* is used, as a positive *r* indicates a positive correlation, and a negative *r* indicates an inverse (yet informative) correlation between a feature and the classifier. Different threshold values were sampled (0.3 , 0.4 , 0.5 , 0.6 , 0.7) and tested in Weka using 10-fold cross-validation. θ = 0.6 proved the highest accuracy (using decision trees) of approximately 93.5%, and resulting in 30 selected features. However, this approach of using the correlation for feature selection is prone to post hoc fallacies, seeing as correlation does not imply causation. 3 features were removed based on the author’s experience in Tennis - reducing the total number of features to 27. Perhaps the most obvious example of such a fallacy would be the umpire. Suppose a player typically wins at a tournament, where a certain umpire happens to referee matches often. If that player plays at a different tournament, where he is less likely to win (i.e. due to a different surface or weather condition), but that same empire referees the match, then the choice of umpire would falsely increase his chances of winning in the algorithm.

Learning algorithm:

- **The training instance:** A training instance consists of the concatenation of two players’ performance data and the classifier value (which player won). The players’ names are kept anonymous as ‘Player1’ and ‘Player2’ during training. This way, the algorithm will learn which performance indicators ensue winning a tennis match.

- **The learner:** Decision trees demonstrated the highest performance (93.5%) out of the examined learning algorithms (Bayes Nets 89.8%, Nearest-Neighbor 88.9%) in terms of accuracy with 10-fold cross validation on the training data. This made decision trees the preferred choice for this learning task, especially considering that the learned hypothesis is easy to interpret - a key feature for such a qualitative learning task . The chosen decision tree algorithm is sci-kit-learn’s optimized implementation of CART (Classification and Regression Trees) which is heavily based on the well known C4.5 algorithm. In contrast to C4.5, CART supports numerical output target variables (regression) and does not compute rule sets but still applies post-pruning to the obtained tree to reduce the overfitting effect. CART constructs binary trees using the feature and threshold that yield the largest information gain at each node. Additionally, limiting the maximum depth, or limiting the minimum number of samples per leaf are measures to reduce the tree’s tendency to overfit to the training data.

‘Tennis Forecast’ - A Machine Learning Project
Full Project Report

Guilherme Klink (guilhermeklink2016@u.northwestern.edu)
Sherif Mostafa (sherifmostafa2015@u.northwestern.edu)
EECS-349-Machine Learning - Northwestern University

- **The prediction instance:** Choose two players and a tournament (yes, in Tennis the player’s performance differs between the tournaments) for which the learner should classify the match outcome. Before being able to predict the match outcome, each player’s performance needs to be extrapolated from his previous performance. An SQL query through the database retrieves the match data (averaged from point-by-point data) for all the matches played over the past 5 years in that specific tournament. The weighted average (obviously, more recent matches are more heavily weighted) of the previous performance metrics is obtained. The feature values of the two players are concatenated and passed-on to the learning algorithm for prediction. It is important in Tennis to look at each of the 4 major tournaments separately, as most top players have favorite tournaments in which they disproportionately dominate compared to other tournaments (surface, weather, psychological motivation, fanbase etc. all play a role). Finally, the learner outputs a prediction of which player is more likely to win the match.

Project results:

The outcome of the project was measured on the prediction accuracy of Roland Garros 2015. Due to the recency of the tournament, no suitable data file containing the results was available. Many of the early round players were first time major tournament contestants, such that no performance prediction could be extrapolated. The predictions were performed manually from the 3rd round onwards, delivering an accuracy of 89% correctly predicted tennis matches.

Deficiencies and extensions:

Given the time constraint, there are some deficiencies and areas of improvement that the developers will be actively working on, even past the deadline of the project submission.

One definite area of improvement would be how the averages are obtained. The current algorithm calculates the performance indicators for a prediction instance from the weighted average of the players’ previous performance. However what it does not account for, is how many matches this average was obtained from. Where this matters the most, is in example a player who played very few, but very good matches (i.e. due to early injury and permanent retirement). Such a player’s stats would be very strong - falsely predicting many victories, although the player is out of shape or had a short-lived lucky streak.

As in any sport, there are unpredictable upsets (occasions where a clearly dominant player loses to an underdog). These can obviously not be predicted by the learner. An example of this is Stanislas Wawrinka beating Roger federer in the 2015 French Open. Roger has a 18-2 track record against Stan, as well as clearly higher performance metrics, but still lost to Wawrinka,

Guilherme Klink (guilhermeklink2016@u.northwestern.edu)
Sherif Mostafa (sherifmostafa2015@u.northwestern.edu)
EECS-349-Machine Learning - Northwestern University

who was playing way above his average level of play.

Furthermore, using this forecaster will only return meaningful results when having 2 players from the same tour (men’s or women’s) play each other. This is due to the fact that the training set consists of real, homogenous matches that are separated for each of the two tours.