Poznań 7 listopada 2008

CASE STUDY 1-4 z przedmiotu „Zaawansowana Eksploracja Danych" – V rok TPD

Analiza danych nt. predykcji kosztów produkcji oprogramowania.

Cel :

„Case study" powinien prowadzić do odkrycia użytecznych i potencjalnie interesujących regularności z rzeczywistych danych. Należy także dokonaniu interpretacji i oceny znalezionych regularności. Zalecane jest interpretowanie znalezionych regularności jako form reprezentacji wiedzy odkrytych w bazie danych. Problem dotyczy analizy danych historycznych o realizacji wybranych projektów programistycznych i poszukiwania modelu predykcji kosztów realizacji projektu na podstawie tzw. wskaźników COCOMO opisujących te projekty. Z metodologicznego punktu widzenia sugeruje się wykorzystywanie poznanych metod eksploracji danych i odkrywania wiedzy w zakresie klasyfikacji i regresji - zarówno statystycznych jak i wywodzących się ze sztucznej inteligencji lub także czysto „data mining"-owych.

Podsumowaniem analizy powinien być raport zawierający listę najbardziej interesujących regularności oraz komentarz lub ich interpretacja - raport ten powinien być tworzony na bieżąco podczas zajęć i oddany na czas.

Wprowadzenie

Dążenie do poprawy jakości produkowanego oprogramowania powiązane jest z wprowadzaniem nowych technik organizacji działań w zespołach programistycznych, rozważaniem nowych praktyk, itp. Prowadzi się także badania pomiarowe nad różnymi aspektami procesu wytwarzania oprogramowania – raczej w odniesieniu do „dużych" projektów o odpowiednim znaczeniu praktycznym (z j. ang. tzw. *software measurement*). Jeśli pomiary prowadzone są właściwie w ramach dobrze zaprojektowanych eksperymentów albo jako są rejestracją działań w rzeczywistych projektach, to mogą one doprowadzić do zgromadzenia historycznych danych referencyjnych, które będą podstawą do głębszej analizy bieżących praktyk oraz identyfikacji czynników wpływających na sukces lub porażkę realizowanych projektów programistycznych. Czytając literaturę należy jednak zauważyć, że zgromadzenie wiarygodnych danych o wystarczającej liczbie projektów programistycznych jest dość trudne i rzadko, kiedy dysponuje się bazami danych o dużych rozmiarach.

Jeden z ważnych problemów badawczych i praktycznych w powyższej dziedzinie jest ***oszacowywanie kosztu produkcji oprogramowania***. Na ogół wiąże się to z przewidywaniem ilości osobo-miesięcy (z j. ang. tzw. *Effort in person-month*) dla danej instancji planowanego projektu. Historycznie od lat 80-tych wprowadza się różne modele teoretyczne mające ułatwić kierownikom projektów wykonanie takich oszacowań. Jednym z nich, nadal aktualnym, jest tzw. COCOMO model (skrót od nazewnictwa angielskiego *COnstructive COst MOdel*) wprowadzony przez Boehm'a. W rzeczywistości zaproponowano kilka podstawowych modeli różniących się stopniem dostępnej wiedzy o możliwościach wykonania projektu, lecz wszystkie one wykorzystują podstawą formułę, gdzie koszt lub pracochłonności mierzona w

osobo-miesiącach, ozn. *E*, jest zależny/a od przewidywanej wielkości tworzonego oprogramowania (w tysiącach linii kodu), ozn. *S*, np. może być ona postaci

$$E = a \cdot S^b \cdot F,$$

gdzie *a*, *b*, *F* to współczynniki zależne od typu tworzonego oprogramowania i warunków realizacji projektu. Pomimo prostoty takie modele nie są łatwe do praktycznego dostrojenia. Ponadto nie dostarczają wytłumaczenia, jakie czynniki wpływają na koszt rozważanego projektu. Dlatego poszukuje się nowych modeli wiążących informacje o współczynnikach mierzonych w trakcie realizacji projektów (lub do oszacowania na podstawie wiedzy o naturze projektu, doświadczeniu zespołu programistycznego, itp.) a przewidywaną wartością kosztu jego wykonania. W tym celu wykorzystuje się także metody analizy i eksploracji danych (osoby zainteresowane mogą wykonać samodzielne studia literaturowe aktualnych publikacji / raportów naukowych na powyższe tematy – można np. skorzystać ze specjalistycznych wyszukiwarek internetowych).

W celu zapoznania się z powyższym problemem zastosowania metod eksploracji danych do oszacowania kosztów produkcji oprogramowania proponujemy wykonanie analizy znanej z bazy danych o realizacji historycznych projektów zebranych oryginalnie przez Boehm'a dla potrzeb modelu COCOMO. Dane dotyczą 63 zakończonych projektów, które opisane są przez 23 atrybuty warunkowe (opisujące tzw. predictive COCOMO factors) i dwa atrybuty decyzyjne (wynikowe). Atrybuty decyzyjne odnoszą się do:

tkdsi - (size) total delivered source instructions, czyli rozmiaru dostarczonego ostatecznie kodu programu,

effort - effort in men/month, czyli ostateczny osobo-koszt projektu (pracochłonność).

Dokładny opis atrybutów warunkowych podany jest w załączniku 1.
Należy zwrócić uwagę, że oryginalnie atrybuty decyzyjne zdefiniowane są na skali liczbowej. W trakcie analizy można problem rozważać jako zadanie:

- ***Regresji***, przewidujemy wartości liczbowe atrybutów wynikowych.
- ***Klasyfikacji***, w tym przypadku konieczna jest dyskretyzacja zakresu zmienności atrybutów decyzyjnych.

W drugim przypadku, można rozważyć różne propozycje dyskretyzacji, przy czym należy zakładać ich sensowną interpretacją, jako zmienną *porządkową* np. niska, średnia, wysoka pracochłonność. Przykładowo dla 3 klas decyzyjnych może być ona następująca:

effort - (0,10);[10,37.5), [37.5, ..]
tkdsi - ≤ 49, [50,170], ($\geq$170,…]

Pamiętaj, że jest to tylko propozycja, możesz samodzielnie poszukiwać innej dyskretyzacji, także stosując inną liczbę przedziałów, przy czym sugerujemy, aby było ich nie mniej niż trzy. W literaturze można także spotkać sugestie użycia jednego zbiorczego atrybutu decyzyjnego zdefiniowanego jako stosunek tkdsi/effort.

Celem analizy dostarczonych danych jest próba odpowiedzi na pytanie takie jak:

- Jakie są związki pomiędzy wartościami atrybutów, a w szczególności między atrybutami warunkowymi a decyzyjnymi?
- Które z atrybutów mają największe znaczenie dla poprawnego oszacowania kosztu/pracochłonności projektów? Czy można określić hierarchię ważności atrybutów?
- Czy można dokonać selekcji / redukcji atrybutów zapewniając równocześnie nie pogorszenie zdolności predykcyjnej budowanego modelu?

- Czy jest możliwe skuteczne przewidywanie wartości wynikowej na podstawie analizy danych historycznych (innymi słowy czy można zbudować skuteczny system predykcyjny lub klasyfikacyjny?

Podsumowując problem, jaki sobie stawiamy dotyczy poszukiwania istotnych zależności, regularności pomiędzy atrybutami, a także ustalania hierarchii ważności atrybutów opisujących projekty oraz oceny błędu predykcji albo trafności klasyfikowania.

Powyższe badanie ma przede wszystkim doprowadzić do powstania raportu naukowego. Może to być wykorzystane także do celów szkolenia studentów. Konieczna jest w miarę formalna analiza i uzasadnianie wniosków.

Powinieneś pamiętać, iż nie masz wpływu na rozmiar dostępnych danych, nie możesz oczekiwać dostarczenia dodatkowych opisów projektów; zostało to wykonane przed Twoim udziałem w studium badawczym; Tzn. nie będziesz miał dodatkowych obserwacji lub wprowadzenia dodatkowych atrybutów. Jeśli jesteś zainteresowany to możesz skorzystać z dodatkowego pliku kemerer – zawierającego informacje o kolejnych 15 projektach, = lecz uważaj nie są one opisywane przez wszystkie atrybuty użyte w podstawowym pliku (cocomo), ponadto oryginalnie zostały zebrane przez innych zespół badawczy niż tworzący plik podstawowy. Przy zachowaniu odpowiedniej ostrożności możesz sprowadzić schematy atrybutów w obu plikach do wspólnej postaci.

Powinieneś także, ocenić jakość dostarczonych danych, zwracając uwagę, na ew. błędy. Ponadto, dane dotyczące projektów mogą charakteryzować się występowaniem obserwacji nietypowych, lub samotniczych czy innymi artefaktami.

Natomiast, jeśli potrafisz ocenić jakość otrzymanych danych możesz dokonywać przeskalowań lub przedefiniować atrybuty (np. tworzyć nowe w oparciu o oryginalnie pomierzone czynniki), jeśli ich końcowa postać jest akceptowalna dla potencjalnego użytkownika (czytaj $\rightarrow$ ma potencjalnie dogodną interpretacje w rozważanym zastsowaniu).

Należy także pamiętać, że dane w tego typu zastosowaniach są raczej małej wielkości, istnieje wiele atrybutów o różnym znaczeniu, sam proces tworzenia oprogramowania i planowania wysiłku jest ciągle opisywany jakościowo, nieprecyzyjnie i potencjalnie „trudny do matematycznego/ilościowego modelowania”, co powinno być uwzględniane w trakcie analizy i interpretacji wyników.

Inne uwagi metodyczne:

- Opłaca się badać jakość dostarczonych danych (mogą być zbierane przez osoby, które nie znają własności Twoich metod);
- Interesujące jest badanie wzajemnych współzależności tkwiących w danych.
- Nie istnieje jasno wyrażony atrybut decyzyjny, co więcej może być ich wiele.
- Warto stosować więcej niż jedną metodę eksploracji danych (ukierunkowanych na różne formy wiedzy i różne ich reprezentacje).

W załącznikach otrzymujesz:

1. Opis problemu wraz interpretacji wartości atrybutów liczbowych.
2. Plik z danymi (format tekstowy arf):

- Podstawowy plik (63 projekty) ma nazwę cocomo,
- Dodatkowy plik (15 projektów) ma nazwę kermerer.

---------- **Załącznik nr 1**

Lista atrybutów opisujących projekty w pliku cocomo (opis z materiałów anglojęzycznych):

ATTRIBUTES CLASSIFICATION ACORDING TO ORIGINAL COCOMO

(Prod) - product atributes (3)

(Comp) - computer attributes (4)

(Pers) - personel attributes (5)

(Proj) - project attributes (3)

(oth) - factors not included in original COCOMO (10)

ATTRIBUTES DETAILED DESCRIPTION

mode - (oth) software development mode [embedded, semidetached, organic]

appl - (oth) type of application [business, control, human-machine, scientific, support, system]

lang - (Prod) language level [Fortran, Cobol, PL1, Pascal, APL, PL/S, Jovial, C, CMS-2]

rely - (Prod) required software reliability

data - (Prod) size of data base

cplx - (Prod) product complexity

aaf - (oth) adaptation adjustment factor (adjustment factor for size instructions that were adopted instead of newly developed)

time - (Comp) execution time constraint

stor - (Comp) main storage constraint

virt - (Comp) virtual machine volatility (virtual machine = hardware and software under which works analysed SW system)

turn - (Comp) computer turnaround time

type - (Comp) type of computer [maxi, midi, mini, micro]

acap - (Pers) analyst capability

aexp - (Pers) applications experience

pcap - (Pers) programmer capability

vexp - (Pers) virtual machine experience

lexp - (Pers) language experience

cont - (Pers) personnel continuity [low, nominal, high]

modp - (Proj) use of modern programming practices

tool - (Proj) use of software tools

sced - (Proj) required development schedule

rvol - (Proj) requirements volatility

DECISION ATTRIBUTES

tkdsi - (size) total delivered source instructions

effort - effort in men/month