Sistem de gradare automat al proiectelor la Programare Orientată pe Obiecte prin modelul Regresiei Liniare

**Realizatori:**

**Stud. sg. maj. Florea Vlad-George**

**Stud. sg. maj. Cujbă Mihai**

**Stud. sg. maj. Nișulescu Alexandru**

# Introducere

Acest document are scopul de a servi drept documentație pentru cel de al doilea proiect dat ca temă la materia „Inteligență Artificială”.

Proiectul reprezintă o soluție software dezvoltată în limbajul de programare Python, versiunea 3.8, ce permite utilizatorului gradarea automată a temelor unor studenți la materia „Programare Orientată pe Obiect” prin folosirea modelului regresiei liniare.

# Fundamente teoretice Regresie Liniară

Regresia liniară reprezintă un instrument matematic ce permite modelarea unei relații liniare între o valoare țintă și mai multe variabile dependente sau independente.

# Regresie Liniară Simplă

În forma ce-a mai simplă regresia liniară ilustrează relația dintre două variabile, spre exemplu: relația dintre media obținută la bacalaureat și nota obținută la examenul de licență.

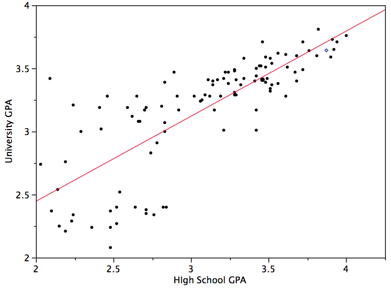


Figura 1: Regresie Liniară Simplă

# Regresie Liniară Multiplă

Acest tip de regresie ilustrează relația liniară dintre mai multe variabile și o valoare scalară rezultat.

Prin intermediul acestui model putem aproxima valoarea scalară ca o combinație liniară a caracteristiciilor. Spre ex.: Modelarea relației dintre prețul unei case și anul în care a fost construită, numărul de camere și zona unde se află.

# Soluția Software

Aplicația dezvoltată reprezintă un instrument software ce permite utilizatorului aproximarea unei note a unui proiect la „Programare Orientată pe Obiecte” folosind modelul Regresiei Liniare Multiplă.

# Modelul Datelor

Pentru a folosi modelul Regresiei Liniare este necesară identificarea valoriilor caracteristice, respectiv valoarea țintă și caracteristiciile.

Valoarea țintă este reprezentată de nota finală a proiectului.

Fiind vorba de un proiect la „Programarea Orientată pe Obiecte” dezvoltat în limbajul C++ au fost identificate următoarele atribute:

* Existența unei diagrame de clase
* Existența unui fișier readme
* Numărul total de clase
* Numărul de clase abstracte
* Numărul de clase normale
* Numărul de interfețe
* Numărul de moșteniri
* Numărul de metode virtuale

# Ipoteza

Ipoteza pe care este bazată aplicația este că există o relație liniară între atributele identificate și nota finală a proiectului.

# Parsarea datelor

Datele neprelucrate vin sub forma unui director ce conține proiectul unui student.

Soluția software extrage atributele fiecărui proiect împreună cu id-ul studentului și le salvează într-un fișier de tip CSV.

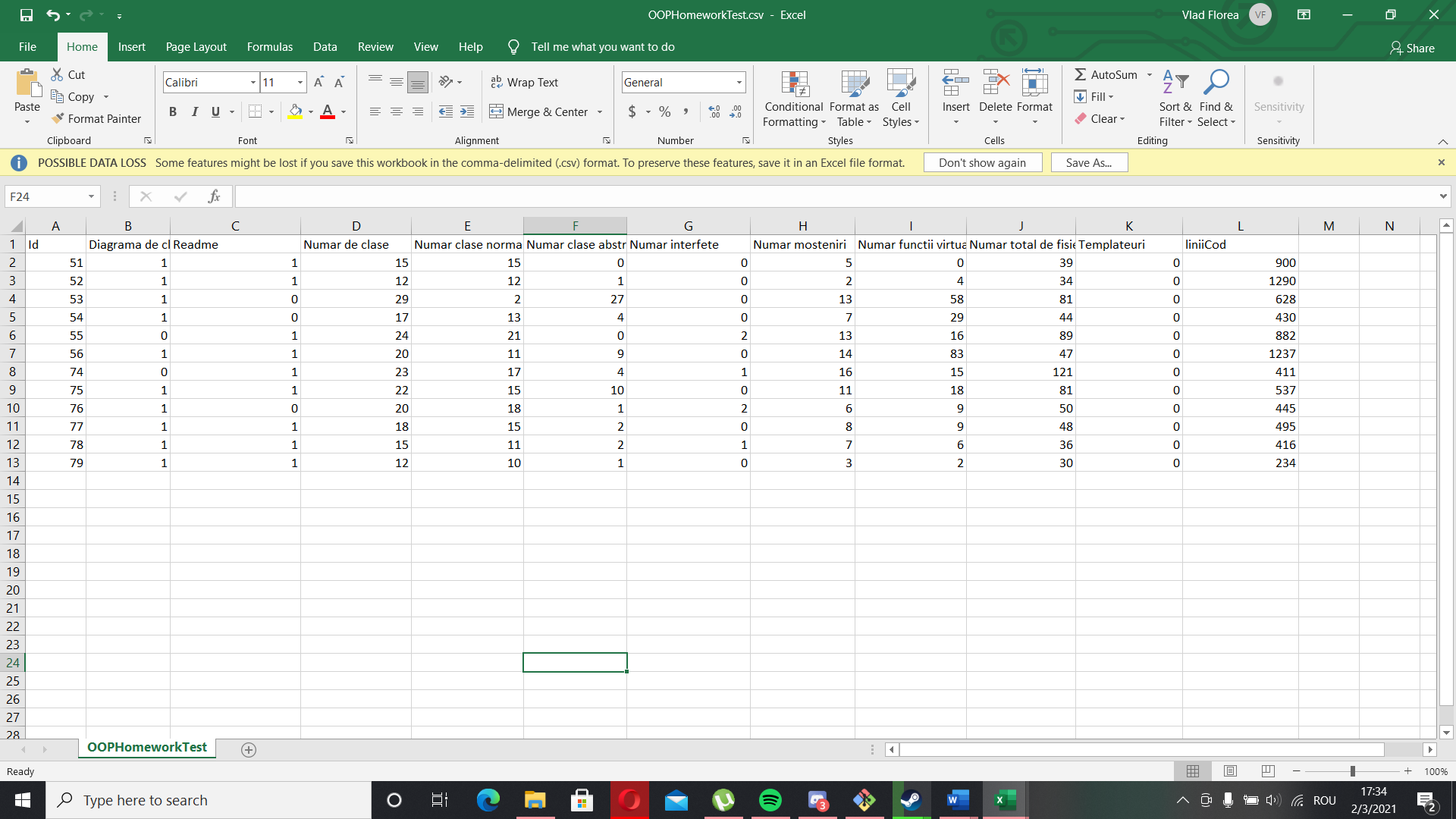


Figura 2: Fișier CSV

# Prelucrarea datelor

Prin intemediul setului de date de antrenare și a fișierului labels.txt se construiește un fișier CSV ce conține atributele fiecărui student împreună cu nota acestuia.

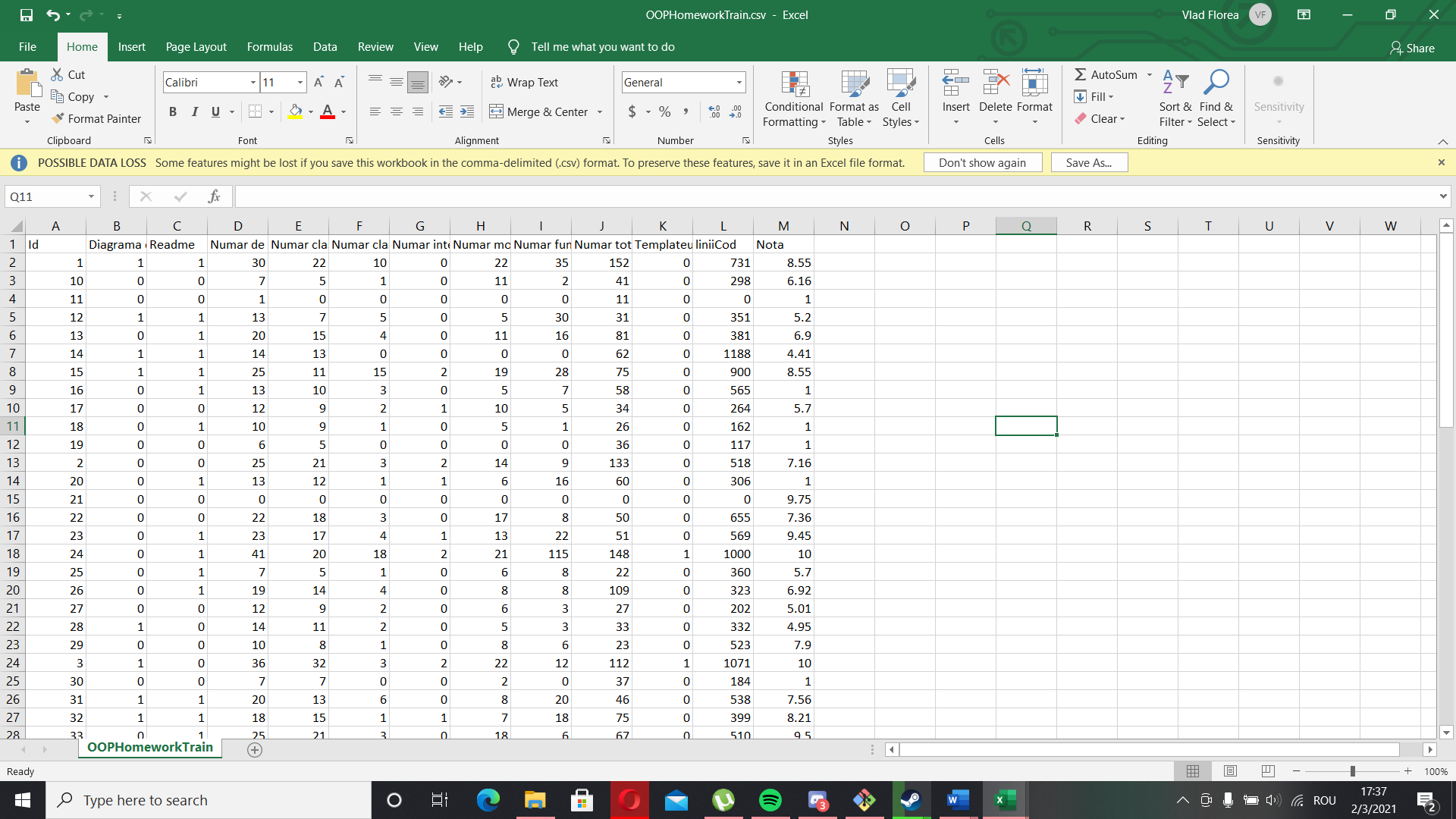


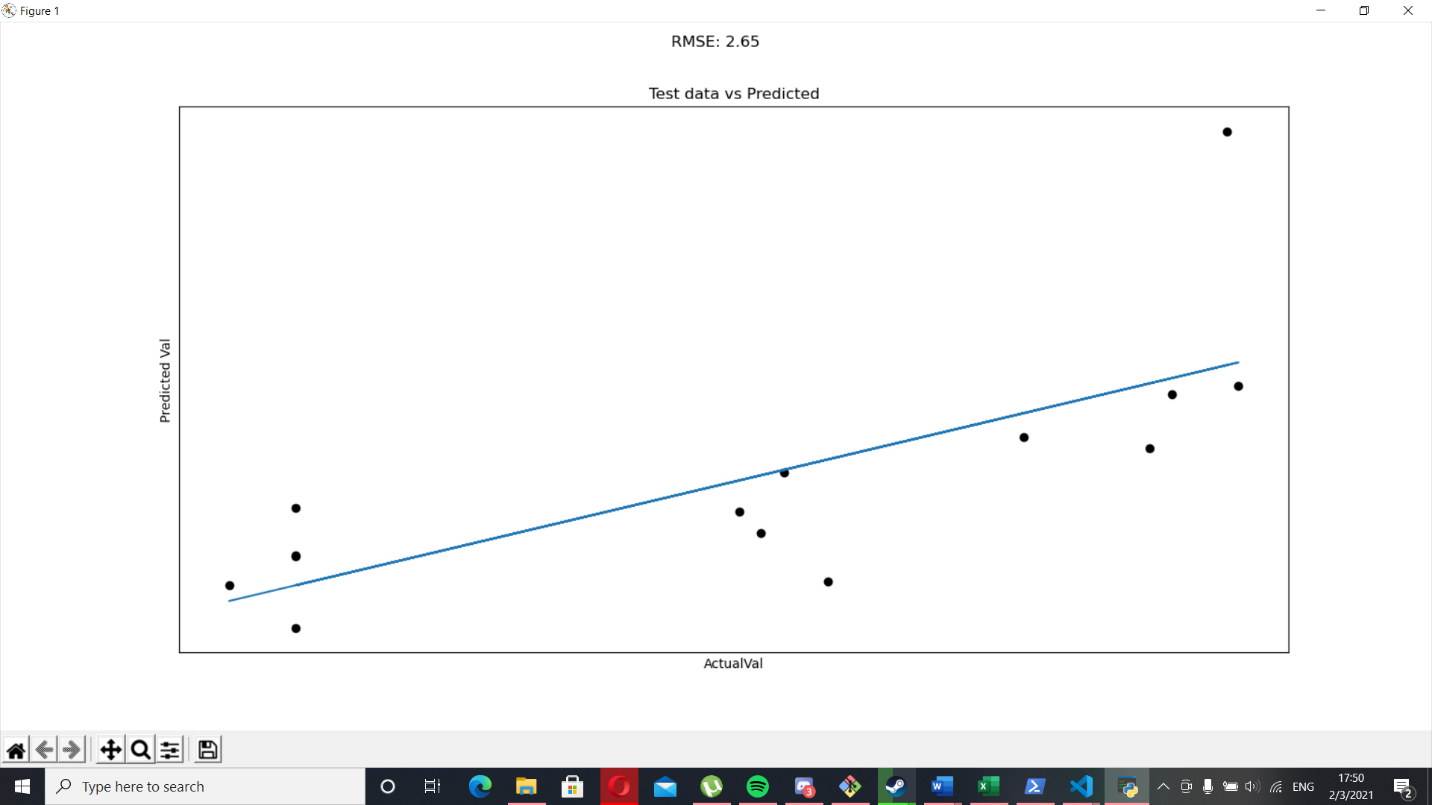
Figura 3: Fișier de antrenare CSV

Folosind librăria Scikit Learn extragem din fișierul CSV atributele identificate și variabila țintă (Nota), scalăm datele și le împărțim în două seturi:

* Setul de antrenare – Train Set – 80% din total
* Setul de testare – Test Set – 20% din total

# Rezultate obținute

După crearea modelului folosind setul de antrenare, verificăm modelul folosind datele de test și comparăm rezultatele prezise de model cu cele din fișierul labels.txt, în urma scalării rezultatelor finale( nota trebuie să fie între 1-10).



RMSE-ul obținut este de 2.65.

# Modulele folosite

Acest capitol este dedicat ilustrării modulelor ce stau la baza aplicației dezvoltate.

# StudentGrader.py

Acest modul este responsabil pentru notarea proiectelor studențiilor prin intermediul modelului de regresie liniară generat.

# Funcția gradeAStudent

Această funcție este folosită pentru a prezice nota unui singur student.

# Funcția gradeACsvOfStudents

Această funcție primește ca input un fișier CSV cu atributele studențiilor și prezice notele fiecăruia, aceasta are și opțiunea de a scrie rezultatele într-un fișier CSV.

# root.py

Conține modelul datelor pentru un student. Un student este reprezentat digital sub forma unei clase care conține atributele identificate în proiectul său.

# projectParser.py

Modulul este responsabil pentru parsarea proiectelor studențiilor și extragerea atributelor dorite.

# Funcția extractStudentInfo

Aceasta funcție generează un obiect de tip Student prin parsarea unui director sursă și verificarea tuturor fișierelor din interior.

# Funcția extractStudentsList

Această funcție genrează o listă de obiecte de tip Student prin parsarea tuturor proiectelor aflate într-un director rădăcină.

# Funcția getFirstLevelSubdires

Funcție ce returnează directoarele de la nivelul inferior (Doar primul nivel).

# parse.py

Aceasta este modulul responsabil pentru obținerea tuturor informațiilor folosite ca metrici referitoare la proiectele studențiilor.

# Funcția count\_virt

Această funcție numără câte funcții virtuale au fost implementate

# Funcția count\_inh

Această funcție numără câte moșteniri au fost implementate.

# Funcția fileCounter

Această funcție numără câte fișiere au fost create.

**3.4.4. Funcția fileCounter**

Aceasta este metoda responsabilă de numărarea tuturor fișierelor aflate în interorul directorului.

**3.4.5. Funcția interface\_virtuals**

Aceasta funcție returnează numarul de clase abstracte, clase normale și interfețe pe care le foloseste studentul în proiectul său.

**3.4.5. Funcția count\_template**

Aceasta este funcția responsabilă cu numărarea de clase în care au fost folosite template-uri.

**3.4.5. Funcția count\_lines**

Aceasta este funcția folosită pentru a ne returna numărul total de linii de cod scrise de student.

# modelRegresie.py

Responsabil pentru generarea modelului de regresie liniar prin folosirea bibliotecii scikit learn.

# Funcția generateModel

Această funcție genrează un model de regresie liniară folosind ca input un fișier csv

# dataSetFactory.py

Modul ce se ocupă cu generarea fișierelor CSV.

Există două tipuri de fișiere CSV:

* Cele de testare- Care nu conțin nota finală. Aceasta trebuie preziză
* Cele de antrenare- Care conțin nota finală și aceasta trebuie extrasă din fișierul labels.txt

# Funcția generateNormalCsv

Funcție ce genrează un fișier CSV de testare

# Funcția generateDatasetCsv

Funcție ce generează un fișier CSV de antrenare.

# csv\_module.py

Modul folosit de dataSetFactory pentru a genera fișiere CSV.

# Rezultate

În urma extragerii atributelor din setul de date oferit pentru testare au fost prezise următoarele note.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Id | Diagrama de clase | Readme | Numar de clase | Numar clase normale | Numar clase abstracte | Numar interfete | Numar mosteniri | Numar functii virtuale | Numar total de fisiere | Templateuri | liniiCod | Nota |
| 51 | 1 | 1 | 15 | 15 | 0 | 0 | 5 | 0 | 39 | 0 | 900 | 3.733641 |
| 52 | 1 | 1 | 12 | 12 | 1 | 0 | 2 | 4 | 34 | 0 | 1290 | 1.486727 |
| 53 | 1 | 0 | 29 | 2 | 27 | 0 | 13 | 58 | 81 | 0 | 628 | 8.23234 |
| 54 | 1 | 0 | 17 | 13 | 4 | 0 | 7 | 29 | 44 | 0 | 430 | 6.396675 |
| 55 | 0 | 1 | 24 | 21 | 0 | 2 | 13 | 16 | 89 | 0 | 882 | 7.570811 |
| 56 | 1 | 1 | 20 | 11 | 9 | 0 | 14 | 83 | 47 | 0 | 1237 | 4.984718 |
| 74 | 0 | 1 | 23 | 17 | 4 | 1 | 16 | 15 | 121 | 0 | 411 | 10 |
| 75 | 1 | 1 | 22 | 15 | 10 | 0 | 11 | 18 | 81 | 0 | 537 | 9.175213 |
| 76 | 1 | 0 | 20 | 18 | 1 | 2 | 6 | 9 | 50 | 0 | 445 | 6.593293 |
| 77 | 1 | 1 | 18 | 15 | 2 | 0 | 8 | 9 | 48 | 0 | 495 | 5.057271 |
| 78 | 1 | 1 | 15 | 11 | 2 | 1 | 7 | 6 | 36 | 0 | 416 | 2.410078 |
| 79 | 1 | 1 | 12 | 10 | 1 | 0 | 3 | 2 | 30 | 0 | 234 | 1 |

# Concluzii

Deși se poate intui o relație între atributele identificate și notele finale, liniaritatea unei astfel de relații nu este evidentă din setul de date oferit.

Este posibilă necesitatea extragerii unor atribute suplimentare, precum:

* Folosirea unor cuvinte cheie specifice paradigmei de programare orientată pe obiect, precum- „Singleton”, „Factory”, etc.
* Identificarea secvențelor de cod ce ilustrează polimorfism, precum:
  + Suprascrierea metodelor
  + Suprascrierea operatorilor
  + Name mangling
  + Etc.
* Prezența erorilor de compilare sau a erorilor la runtime.
* Existența mecanismelor de tratare a excepțiilor.

Totodată, este posibil ca setul de date oferit să fie prea mic.