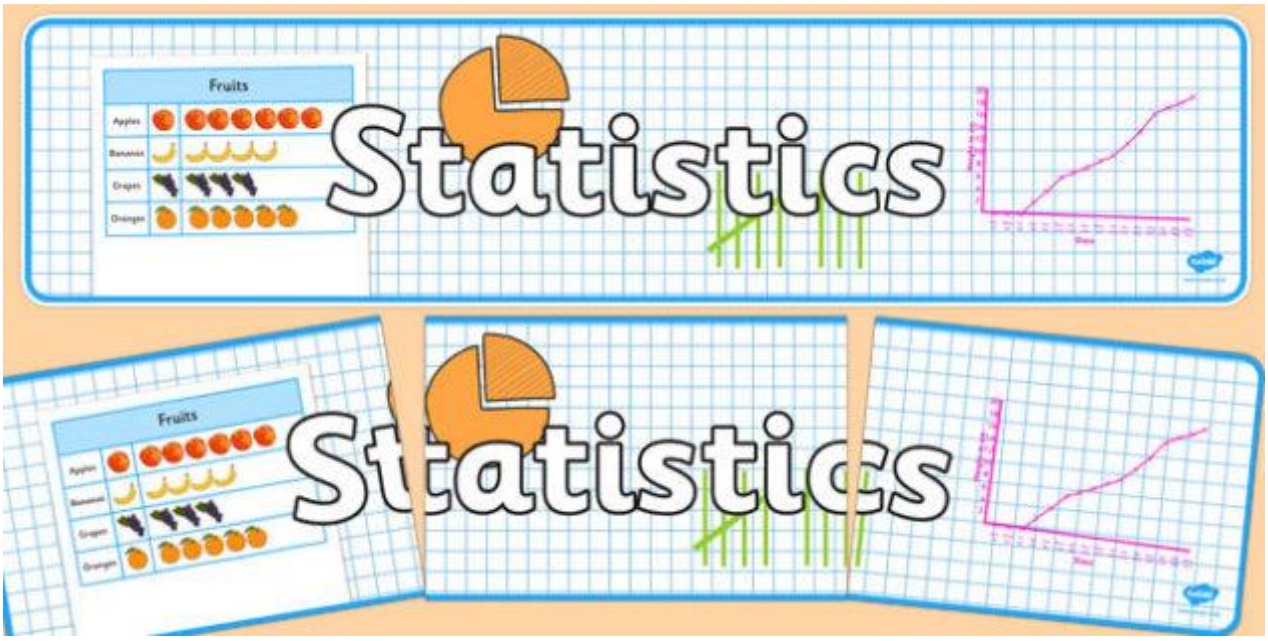


Project môn học

Lập trình nâng cao trong khoa học dữ liệu

Nhóm thực hiện

Tạ Văn Nhân
Nguyễn Hoàng Nam
Phạm Trung Hiếu
Nguyễn Quang Huy

Đề tài

Ứng dụng python trong tự động mô hình hóa thống kê

Repository

https://github.com/namnhGru/auto-modelling-ds-vnu.git

Table of Contents

I. Mở đầu 3
II. Nội dung dự án 4
1. Thiết kế 4
a. Chức năng từng class 5
2. Hoạt động 9
III. Kết quả thực hiện 10
IV. Đánh giá ưu nhược điểm 12
V. Những mục chưa làm được 12
VI. Lời cảm ơn 13

I. Mở đầu

Thưa bạn đọc,

Trong thời gian học tập tại HUS, chuyên ngành khoa học dữ liệu, nhóm nghiên cứu nhận thấy rằng việc mô hình hóa thống kê không chỉ đơn thuần giải quyết về mặt toán học, mà còn là những thao tác dữ liệu trên phần mềm với hiệu năng cao, đem lại nhiều lợi ích trong tính toán.

Đặc biệt, nhóm nhận thấy ngôn ngữ Python hay các ngôn ngữ lập trình nói chung có khả năng xây dựng nên các kịch bản cũng như tạo ra các sản phẩm mang tính nhất quán, giúp giảm thiểu nhiều thao tác lặp lại khi tính toán trên số liệu, cụ thể là những kiểm định, ước lượng thống kê hay các thao tác đọc ghi dự đoán dữ liệu thông thường. Nhờ có Python, các thao tác trên dữ liệu của nhóm đã nhất quán hơn cũng như tránh bị lặp lại ở thời điểm tương lai

Dự án này là một nỗ lực trong việc tổng hợp các thao tác trên dữ liệu lặp như vừa nêu, cũng như là tiền đề để nghiên cứu những ứng dụng và công nghệ cao hơn trong việc tự động hóa mô hình cũng như học máy, giảm các tải phải bỏ ra của các nhà khoa học dữ liệu để đem lại sản phẩm một cách nhanh và chính xác nhất. Dự án cũng hướng đến như một "hộp đen" xử lý dữ liệu, giúp cho những người sắp bước vào con đường dữ liệu có một công cụ mô tả được quá trình xử lý đơn giản nhất, có được cái nhìn tổng quan nhất khi theo đuổi ngành khoa học dữ liệu.

Hà Nội, 04/6/2019,

Nhóm dự án ứng dụng Python trong tự động mô hình hóa

Tạ Văn Nhân, Nguyễn Hoàng Nam,
Phạm Trung Hiếu, Nguyễn Quang Huy

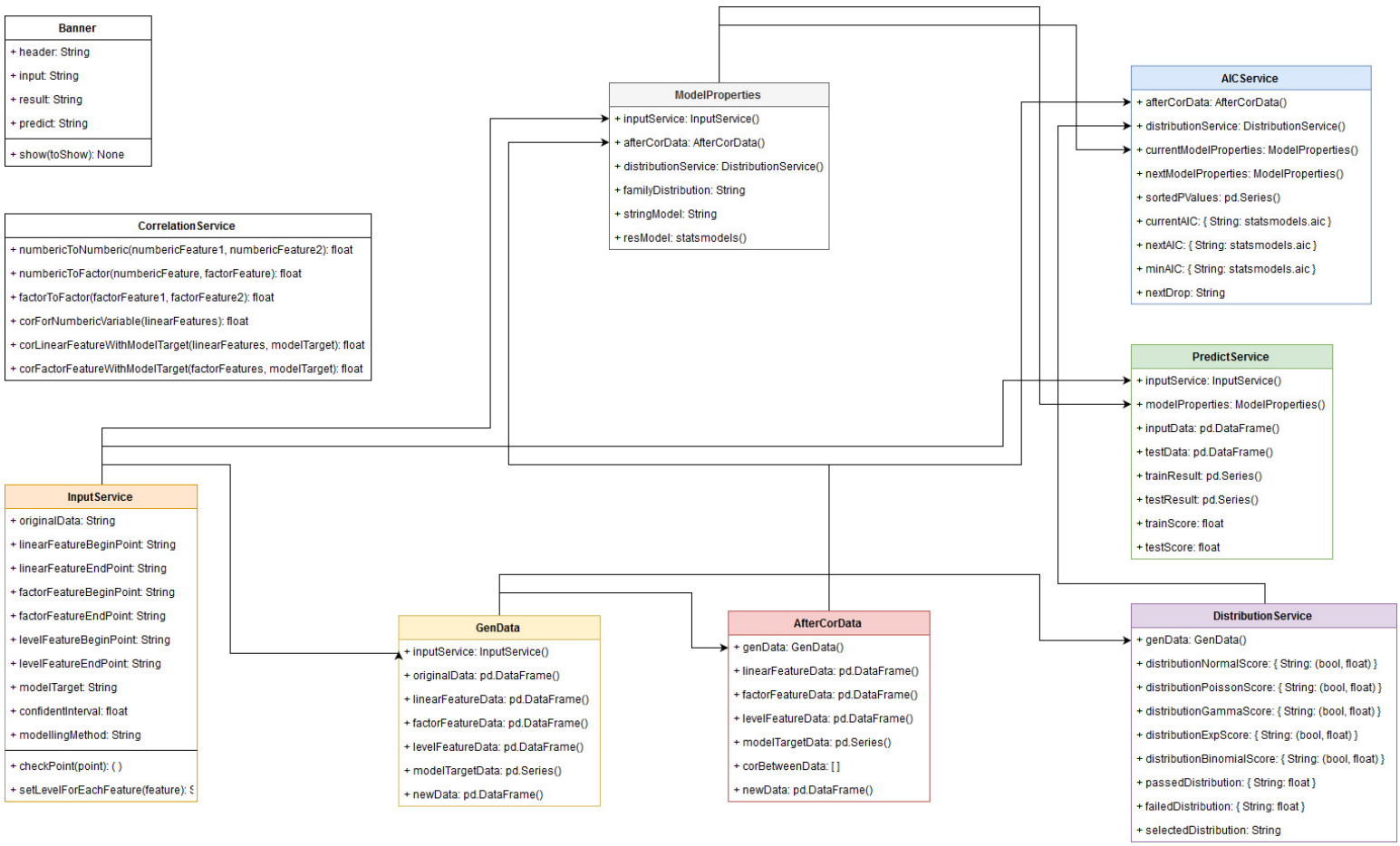
II. Nội dung dự án

1. Thiết kế

Dự án được thiết kế với các mục tiêu sau:

- Tối thiểu nhân lực cho việc triển khai coding
- Code có khả năng sử dụng lại khi cần thiết
- Code có khả năng mở rộng khi cần thiết
- Code có khả năng bảo trì dễ dàng khi cần thiết

Với mục tiêu đó, dự án được thiết kế như sau:



Hình 1: Thiết kế class trong dự án

a. Chức năng từng class

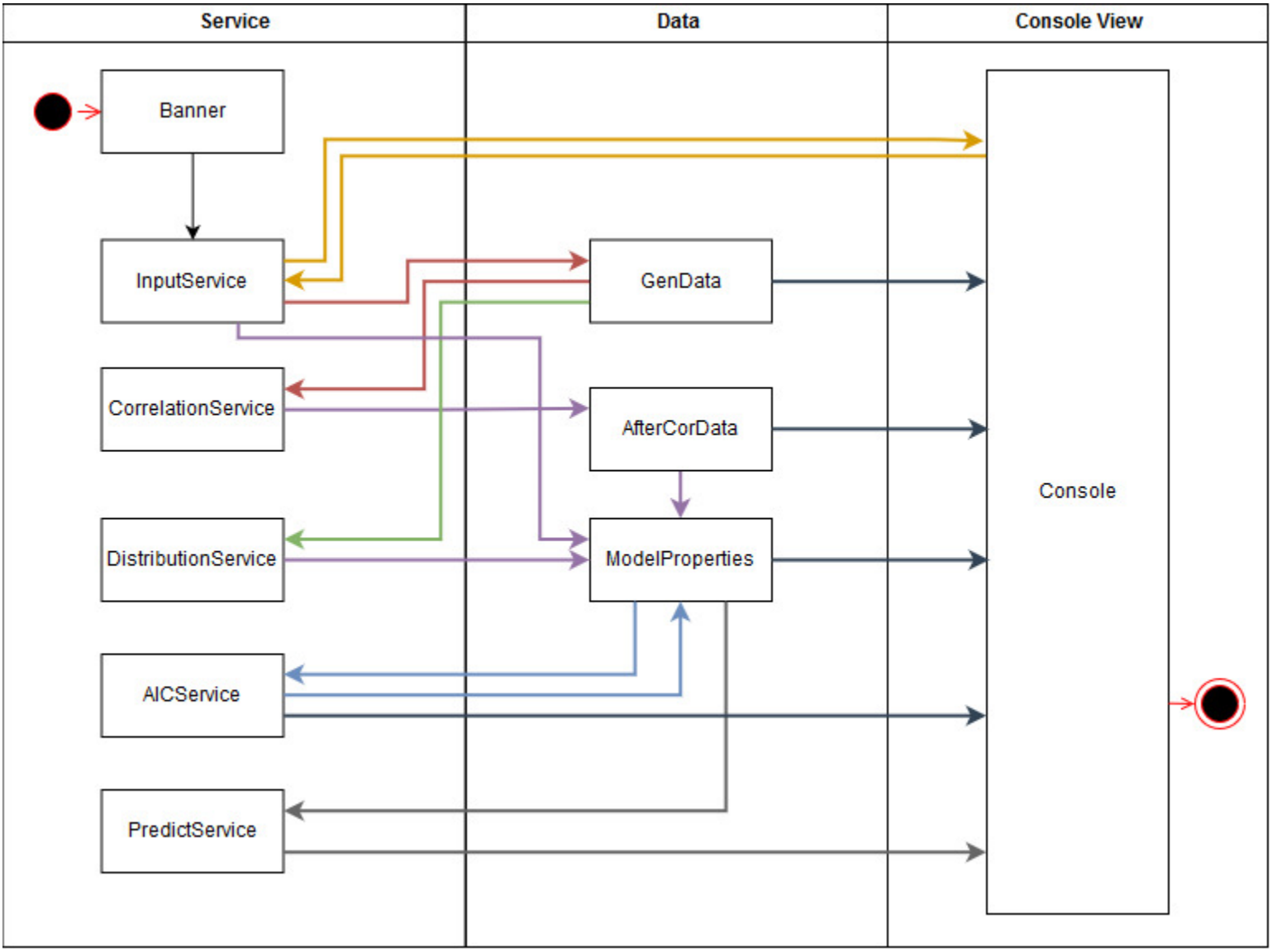
#	Class	Mục đích	Trường	Phương thức
1	`InputService`	• Tiếp nhận Input từ người dùng. • Khối này không hoặc có ít nhất có thể tính năng xử lý các đầu vào người dùng nhập	1. **`OriginalData`**: nhận đường dẫn dữ liệu 2. **`linearFeatureBegin/EndPoint`**: nhận khoảng dữ liệu với dạng đặc trưng là tuyến tính 3. **`factorFeatureBegin/EndPoint`**: nhận khoảng dữ liệu với dạng đặc trưng là category đều 4. **`levelFeatureBegin/EndPoint`**: nhận khoảng dữ liệu với dạng đặc trưng là category phân cấp 5. **`modelTarget`**: nhận dữ liệu quy định là response của mô hình 6. **`confidentInterval`**: nhận độ tin cậy trên toàn dự án 7. **`modellingMethod`**: nhận phương thức lên mô hình (lm hay glm)	1. **`CheckPoint`**: nhận input String khoảng dữ liệu, trả về cận của khoảng dữ liệu 2. **`SetLevelForEachFeature`**: nhận input string category phân cấp, trả về thứ tự phân cấp 3. Ngoài ra là các phương thức get/ set xử lý cho trường
2	`GenData`	• Xử lý các input và trả về bảng dữ liệu đã được quy định trường và thuộc tính	1. **`InputService`**: nhận object **`InputService`** để lấy dữ liệu 2. **`originalData`**: dữ liệu gốc đã được xử lý 3. **`linearFeatureData`**: dữ liệu tuyến tính đã được xử lý 4. **`factorFeatureData`**: dữ liệu category đều đã được xử lý 5. **`levelFeatureData`**: dữ liệu category phân cấp đã được xử lý 6. **`modelTargetData`**: dữ liệu response đã được xử lý 7. **`newData`**: bảng dữ liệu mới đã được xử lý	Các phương thức get/ set xử lý dữ liệu cho trường
3	`AfterCorData`	• Lấy các dữ liệu đã xử lý corTest, ANOVATest, trả về bảng dữ liệu thu	1. **`genData`**: nhận object **`GenData`** để lấy dữ liệu cần xử lý 2. **`linearFeatureData`**: dữ liệu tuyến tính đã được xử lý corTest	Các phương thức get/ set xử lý dữ liệu cho trường

		gọn/ bổ sung sau khi loại bỏ các trường không liên quan và sinh các trường mới tới response	3. `factorFeatureData`: dữ liệu category đều đã được xử lý ANOVA Test 4. `levelFeatureData`: dữ liệu category phân cấp đã được xử lý ANOVA Test 5. `modelTargetData`: dữ liệu response đã được xử lý 6. `corBetweenData`: dữ liệu interact giữa các đặc trưng 7. `newData`: bảng dữ liệu mới đã được xử lý	
4	`CorrelationService`	• Xử lý CorTest, ANOVA Test cho dữ liệu	Không có	1. `numbericToNumberic`: xử lý cor giữa các biến dạng tuyến tính 2. `numbericToFactor`: xử lý cor giữa các biến dạng category và dạng tuyến tính 3. `factorToFactor`: xử lý cor giữa các biến dạng category 4. `corForNumbericVariable`: xử lý cor giữa các feature tuyến tính 5. `corLinearFeatureWithModelTarget`: xử lý cor giữa các feature tuyến tính và response 6. `corFactorFeatureWithModelTarget`: xử lý cor giữa các feature category và response
5	`DistributionService`	• Tìm phân phối cho response thông qua các kiểm định	1. `genData`: nhận object GenData để lấy dữ liệu response	Các phương thức get/ set xử lý dữ liệu cho trường

			2. `selectedDistribution`: phân phối của response 3. `passedDistribution`: các phân phối sau kiểm định thỏa mãn 4. `failedDistribution`: các phân phối sau kiểm định không thỏa mãn 5. `Distribution"X"Score`: điểm kiểm định quyết định tính thỏa mãn của các phân phối	
6	`ModelProperties`	• Xử lý việc lên model cho dữ liệu	1. `inputService`: nhận object `InputService` để lấy dữ liệu chọn mô hình 2. `afterCorData`: nhận object `AfterCorData` để lấy dữ liệu đã điều chỉnh sau kiểm định tương quan 3. `distributionService`: nhận object `DistributionService` để lấy dữ liệu phân phối của response 4. `familyDistribution`: xác định link function nếu mô hình được chọn là glm 5. `stringModel`: biểu thức của mô hình 6. `resModel`: mô hình được sinh ra	Các phương thức get/ set để xử lý dữ liệu cho trường
7	`AICService`	• Lựa chọn mô hình tốt nhất từ mô hình đã sinh	1. `afterCorData`: nhận object `afterCorData` để xử lý 2. `distributionService`: nhận object `distributionService` để xử lý 3. `currentModelProperties`: nhận object `ModelProperties` từ mô hình đã tính 4. `nextModelProperties`: nhận object `ModelProperties` sinh ra sau mỗi lần so sánh AIC	Các phương thức get/ set để xử lý trường

			5. **sortedPValues**: nhận danh sách thứ tự các pvalue của các đặc trưng 6. **nextDrop**: tên đặc trưng sẽ loại bỏ tiếp theo trong mô hình 7. **currentAIC**: AIC của mô hình hiện tại 8. **nextAIC**: AIC của mô hình tiếp theo 9. **minAIC**: AIC của mô hình tốt nhất	
8	**PredictService**	• Xử lý dự đoán dữ liệu train và test	1. **inputService**: nhận object **InputService** để lấy thông tin dữ liệu đầu vào 2. **modelProperties**: nhận object **ModelProperties** để lấy thông tin mô hình tốt nhất 3. **testData**: dữ liệu kiểm định 4. **trainResult**: dữ liệu train sau khi dự đoán 5. **testResult**: dữ liệu test sau khi dự đoán 6. **trainScore**: điểm dự đoán dữ liệu train 7. **testScore**: điểm dự đoán dữ liệu test	Các phương thức xử lý get/ set cho trường
9	**Banner**	• Đề mục, chỉ mục cho hiển thị trên console	1. **headerBanner**: Đề mục đầu trang 2. **inputBanner**: chỉ mục nhập dữ liệu 3. **resultBanner**: chỉ mục hiển thị kết quả 4. **predictBanner**: chỉ mục hiển thị dự báo	toShow: hiển thị banner tùy ý

2. Hoạt động



Hình 2: Dòng chảy hoạt động của project

Hoạt động của sản phẩm được mô tả kỹ càng như trên hình vẽ, với các flow cùng màu là cùng một quá trình vào ra của dữ liệu cũng như xử lý. Bạn đọc có thể bắt đầu từ điểm Begin ở trên cùng bên trái Service Layer, lần theo dấu mũi tên màu đến điểm End ở dưới cùng bên phải ConsoleView Layer để kết thúc flow của sản phẩm

III. Kết quả thực hiện

```
    x1     x2     x3  x4    x5  x6  x7  x8      y
0  200  24420  29600  28  47.0   1   0   1  71400
1  228  19993  32582  29  28.0   1   0   1  65200
2  392   4300   4300  22   0.0   0   0   0   7100
3   90  11140  11140  29  10.0   1   0   0  31000
4  126  33060  33060  28  60.0   1   1   0  87000
Column to use as Linear Feature (a->b, leave if none): x1->x5
Column to use as Factor Feature (a->b, leave if none): x6->x8
Column is used as Model Target: y
```

Hình 3: InputService thực thi

```
# Drop feature x5 base on cor.test because p-value is 0.1456940187940498

# Drop feature x6 base on ANOVA test because p-value is 0.5096295136836193

# Drop feature x7 base on ANOVA test because p-value is 0.3029878500868377

# Drop feature x8 base on ANOVA test because p-value is 0.14588500655254666

# Generate feature x1 * x4 base on cor.test between features because p-value is 0.0007499068152671159

# Generate feature x2 * x3 base on cor.test between features because p-value is 1.795880941446326e-05
```

Hình 4: CorrelationService thực thi

```
# Distribution of Model Target is dict_keys(['Normal']) because of p-value is dict_values([0.9917640998036465]) when do selecting between 3 passed distributions (name:
 p-value): {'Normal': 0.9917640998036465, 'Exp': 0.3628463833702418, 'Gamma': 0.9881975132437476}

# 2 false distribution are (name: p-value): {'Poisson': 6.801867518768656e-05, 'Binomial': 0.0}
```

Hình 5: DistributionService thực thi

```
# Model Summary

                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                      y   No. Observations:                   18
Model:                            GLM   Df Residuals:                       11
Model Family:                Gaussian   Df Model:                            6
Link Function:               identity   Scale:                      5.1319e+07
Method:                          IRLS   Log-Likelihood:                -180.89
Date:                Sat, 06 Apr 2019   Deviance:                   5.6451e+08
Time:                        11:18:11   Pearson chi2:                 5.65e+08
No. Iterations:                     3   Covariance Type:             nonrobust
==============================================================================================
                                 coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------------------
Intercept                   6174.3802   3.76e+04      0.164      0.869   -6.74e+04    7.98e+04
newData['x1']               -224.0420    125.477     -1.786      0.074    -469.972      21.888
newData['x2']                 -3.6563      1.013     -3.608      0.000      -5.642      -1.670
newData['x3']                 -0.6394      0.543     -1.177      0.239      -1.704       0.425
newData['x4']               2182.1975   1175.853      1.856      0.063    -122.431    4486.826
newData['x1']:newData['x4']    6.4807      4.404      1.472      0.141      -2.150      15.112
newData['x2']:newData['x3']    0.0002   3.13e-05      4.987      0.000    9.48e-05       0.000
==============================================================================================
```

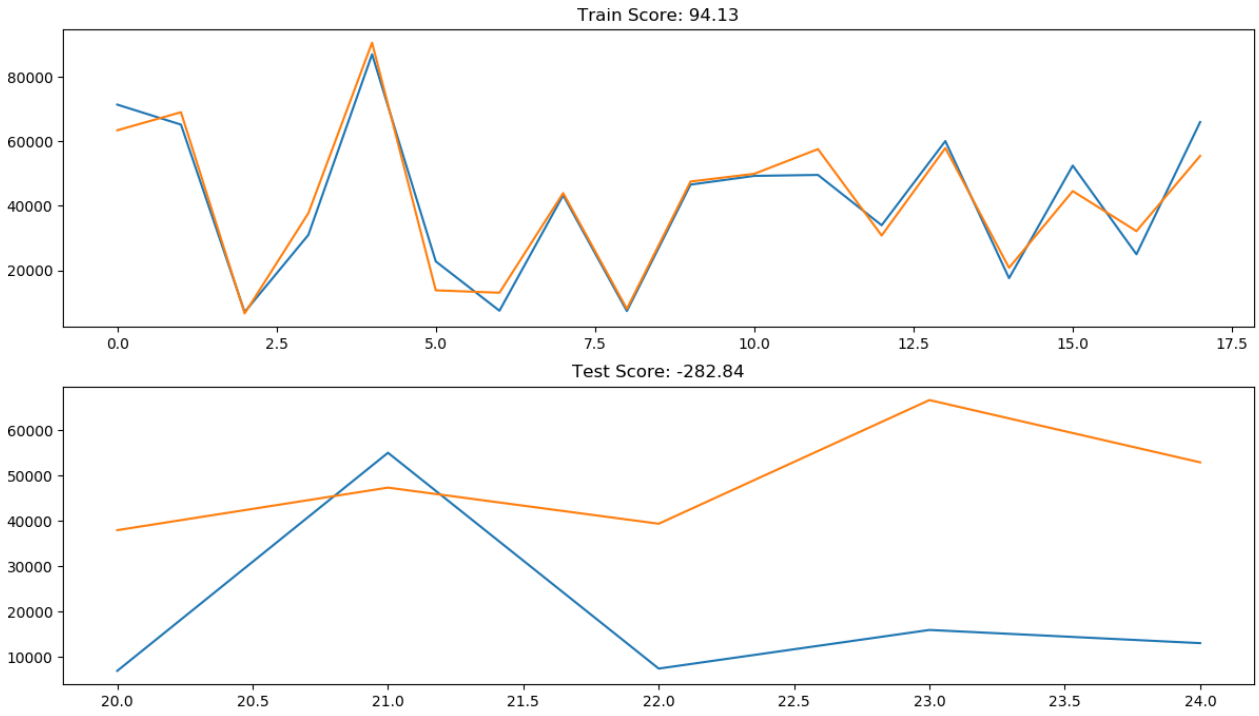
Hình 6: ModelProperties thực thi

```
*** Current Model & AIC: {"y ~ newData['x1'] + newData['x2'] + newData['x3'] + newData['x4'] + newData['x1']:newData['x4'] + newData['x2']:newData['x3']": 375
 .7815457501895}

*** Next Model & AIC: {"y~newData['x1'] + newData['x2'] + newData['x3'] + newData['x4'] + newData['x1']:newData['x4'] + newData['x2']:newData['x3']": 375.7815457501895}

*** Min Model & AIC: {"y~newData['x1'] + newData['x2'] + newData['x3'] + newData['x4'] + newData['x1']:newData['x4'] + newData['x2']:newData['x3']": 375.7815457501895}
```

Hình 7: AIC Service thực thi

```
# ======PREDICT====== #

Train Score: 94.13
Test Score: -282.84
```

Hình 8: PredictService thực thi



Hình 9: Đồ thị dự đoán

IV. Đánh giá ưu nhược điểm

Ưu điểm	Nhược điểm
Loại bỏ các thao tác lặp thông thường của nhà phân tích dữ liệu như đọc dữ liệu, kiểm định thống kê	Phụ thuộc vào 1 kịch bản chung
Phù hợp để làm quy trình sản xuất phần mềm	Logic code phức tạp
Kết quả mô hình được tính toán trong thời gian ngắn	Bảo trì khó khăn vì phải phiên dịch từ thuật toán toán học sang ngôn ngữ phần mềm phụ thuộc vào thiết kế
	Kịch bản kiểm định còn hạn chế

V. Những mục chưa làm được

1. Xử lý trường category phân cấp chưa được ổn định nên chưa đưa vào flow
2. Chưa thử nghiệm đủ nhiều mô hình để đánh giá hiệu năng
3. CorrelationService đang có vấn đề, tuy nhiên là một module quan trọng nên hiện vẫn để trong flow
4. Chưa có giao diện đơn giản cho việc nhập liệu để tránh lặp lại

VI. Lời cảm ơn

Trong quá trình xây dựng sản phẩm, nhóm đã gặp không ít khó khăn về công nghệ, tuy nhiên nhờ sự chỉ bảo tận tình của thầy Vũ Tiến Dũng và giảng viên Nguyễn Tiến Hưởng mà đã có thể tương đối hoàn thành sản phẩm. Xin cảm ơn thầy và bạn rất nhiều.

Ngoài ra, những khó khăn về kiến thức thống kê khi làm sản phẩm cũng được các thành viên trong lớp giải đáp nhiệt tình, xin được cám ơn các bạn.

Lời cuối, cám ơn các thành viên trong nhóm dự án đã không kể thời gian, công việc cá nhân cố gắng hoàn thành dự án đúng hạn bảo vệ.

Xin cảm ơn!

Hà Nội, 04/6/2019,

Nhóm dự án ứng dụng Python trong tự động mô hình hóa