Metody eksploracji danych
Laboratorium
Tematy 1 i 2

Weka + Python + regresja

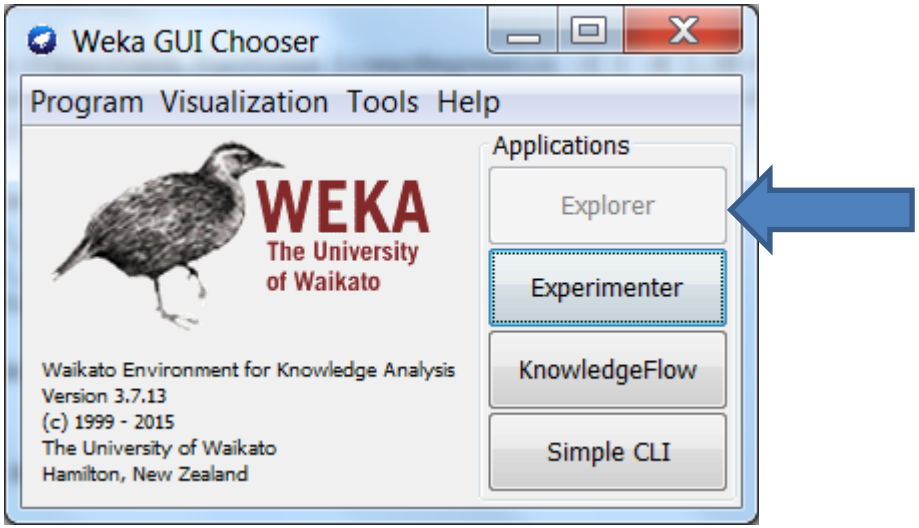
Metody eksploracji danych

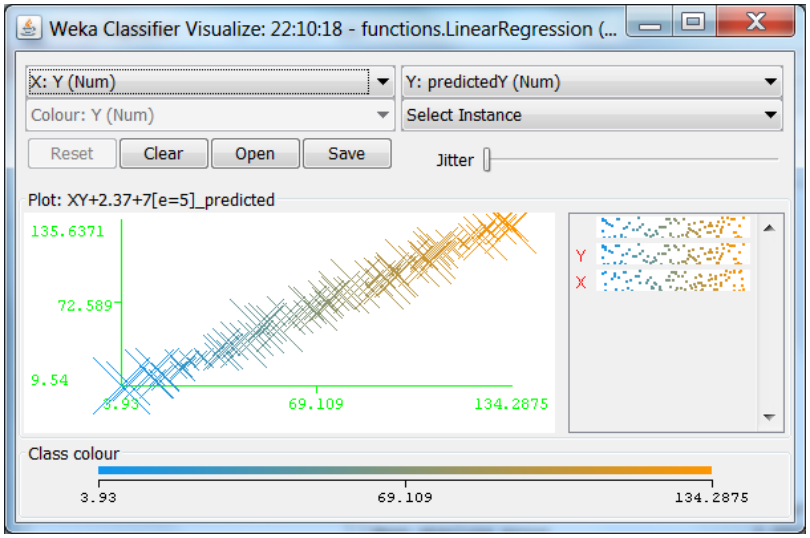
- Zasoby
  - Weka (gdzieś na dysku)
  - Środowisko dla języka Python (Spyder, Jupyter, gdzieś na dysku)
  - Zbiory danych (dostępne na stronie http://home.agh.edu.pl/~pszwed/wiki/doku.php?id=metody_eksploracji_danych)
- Cel
  - Poznanie możliwości Weki
  - Wykonanie klika przykładów regresji
  - Alternatywne środowisko i biblioteki języka Python
  - Typowy workflow podczas analizy danych

Część 1 – Nawigacja po GUI Weka + nieco bibliotek Pythona

1.1 W oknie startowym Weka uruchom opcję Explorer

- Załaduj plik **xy-001.arff** w zakładce **Preprocess**
- Możesz wyświetlić dane w tabeli **Edit**
- Wyświetl zawartość danych w zakładce **Visualize**
- W zakładce **Classify** wybierz **Choose**: functions -> LinearRegression
- Naciśnij Start. Powinno się pojawić równanie prostej
- Niestety za pomocą Weka nie można wyświetlić prostej, co najwyżej zakres błędów klasyfikacji.
- Kliknij prawym klawiszem na result list i wybierz jedną z opcji wizualizacji.





1.2. Wyświetlamy dane i krzywą

Korzystamy z bibliotek w języku Python

1. Używamy IDE Spyder lub Jupyter. Można też przygotować plik tekstowy i uruchomić z poziomu konsoli (mając nadzieję, że biblioteki są zainstalowane). Czasem też działa https://try.jupyter.org/
2. Kopiujemy fragment kodu ze strony: http://home.agh.edu.pl/~pszwed/wiki/doku.php?id=metody_eksploracji_danych
3. Wpisujemy dane w osobnych liniach. """ to specjalny zapis dla tekstu wprowadzanego w wielu wierszach.
4. x, y = np.loadtxt(inp, delimiter=',', usecols=(0, 1), unpack=**True**,skiprows=6) – ładuje tekst z kolumn 0 i 1 do tablic x,y
5. plt.scatter(x,y,s=80, marker='+') – rysuje punkty
6. fx=np.linspace(-10,60,100) 100 równomiernie rozłożonych punktów w zakresie [-10,60]
7. fy=2.3702*fx+6.1973 - wartości funkcji regresji (to są operacje na wektorach)
8. ftrue=2.37*fx+7 – „prawdziwa” funkcja użyta do generacji danych
9. plt.plot(fx,fy,linewidth=2,color='r') – wykres liniowy
10. plt.xlim(-10,60) – zakres wyświetlanych zmiennych

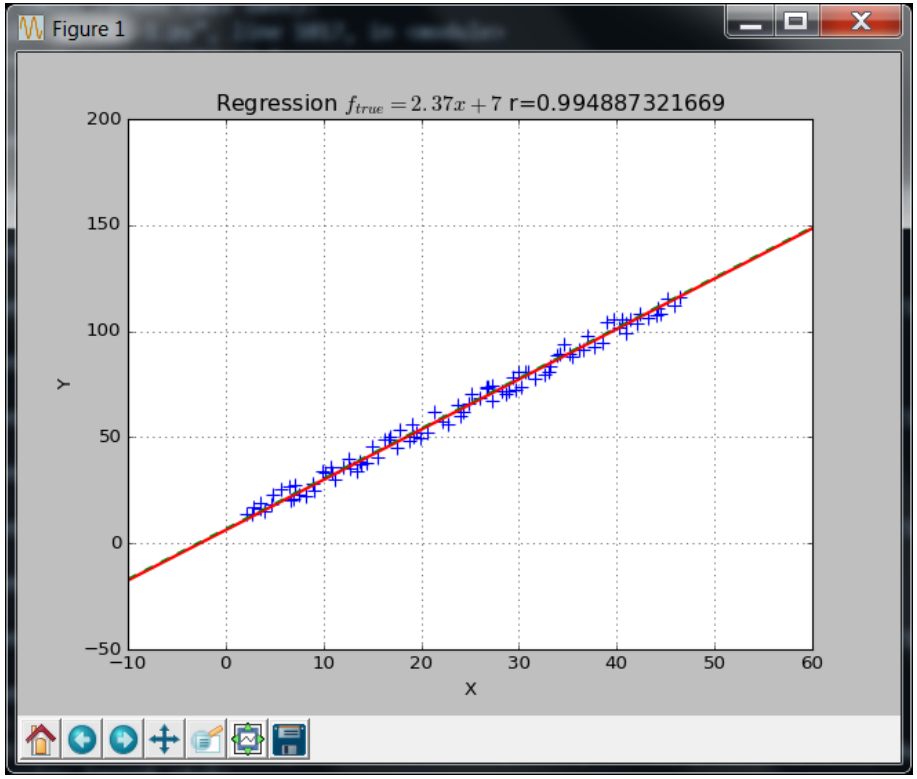
Oczekiwany wynik

- Wykres może być osadzony na stronie HTML w Jupyter: usuń komentarze z pierwszej linii: *%matplotlib notebook*
- Wykres można zapisać w jednym z formatów (bitmapowych/wektorowych)

Wartość r która jest wyświetlana dodatkowo?
Jest to współczynnik korelacji.

Przeczytaj tekst i obejrzyj przykłady:
https://en.wikipedia.org/wiki/Correlation_and_dependence

Jeśli r ≈ 0, zazwyczaj równanie regresji będzie miało postać:
$y = 0x + c$



Inne pliki

Analogicznie załaduj pliki,wyznacz równanie regresji, oceń błędy i wyświetl przebieg krzywej regresji dla:

1.3 Pliku xy-002.arff

1.4 Pliku xy-003.arff

1.5 Pliku xy-005.arff

1.6 Pliku xy-006.arff

1.7 Pliku xy-007.arff

1.8 Pliku xy-008.arff [Skomentuj przebieg krzywej regresji]

1.8 Pliku xy-009.arff

1.9 Pliku xy-010.arff

Zajrzyj do 1.10. Może będzie bardziej efektywnie?

1.10 Realizacja za pomocą bibliotek Python

Najprostsza forma ładowania pliku arff

inp = "xy-001.arff„

x, y = np.loadtxt(inp, delimiter=',', usecols=(0, 1), unpack=True,skiprows=6)

Liczba wierszy do przeskoczenia zależy od pliku

Tworzenie macierzy cech

features=x.reshape(x.size,1)

Musi być to macierz dwuwymiarowa $m \times 1$ (w ogólnym przypadku $m \times n$)

Dopasowanie krzywej

regr = linear_model.LinearRegression()

regr.fit(features, y)

Parametry modelu (regr.coef_ jest tablicą)

print('Coefficients: ', regr.coef_, ' Intercept: ',regr.intercept_)

…

fy= regr.coef_[0]* fx + regr.intercept_

Metody regresji

Metoda najmniejszych kwadratów

```
regr = linear_model.LinearRegression()
regr.fit(features, y)
```

Regresja grzbietowa: metoda gradientu prostego z regularyzacją współczynników, norma L^2

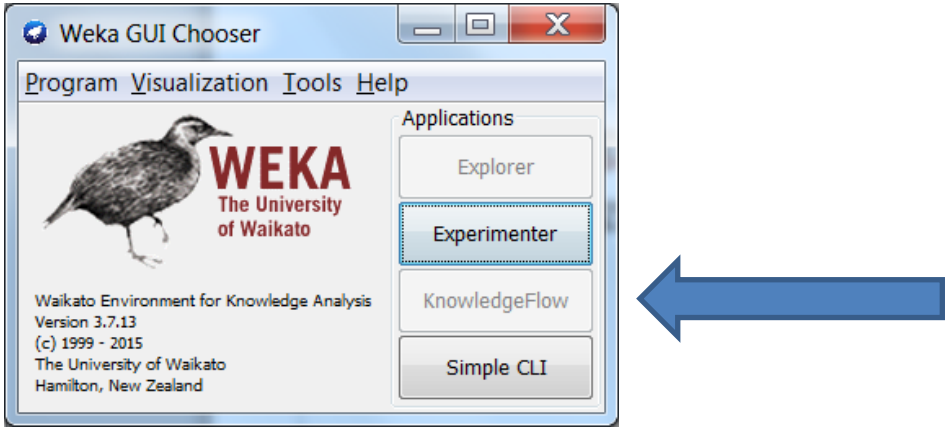
```
regr = linear_model.Ridge (alpha = .5)
regr.fit(features, y)
```

Lasso: metoda gradientu prostego z regularyzacją współczynników, norma L^1

```
regr  = linear_model.Lasso(alpha=0.1)
regr.fit(features, y)
```

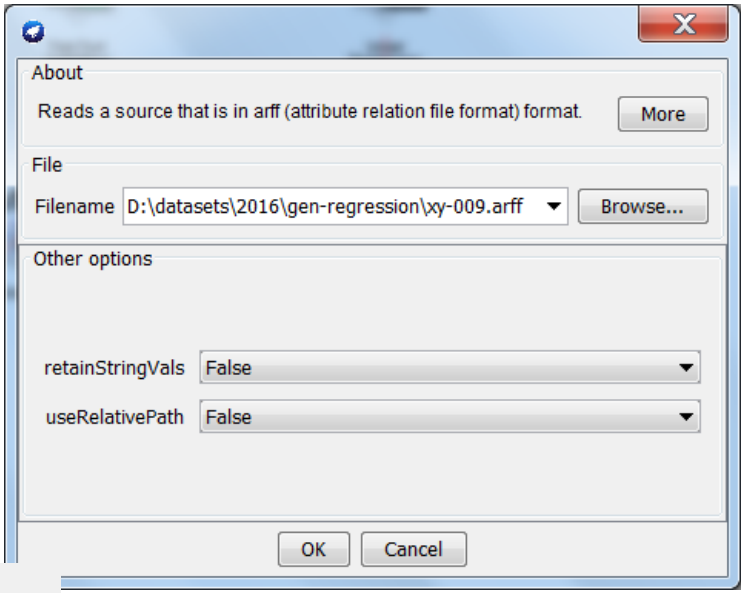
Część 2 KnowledgeFlow

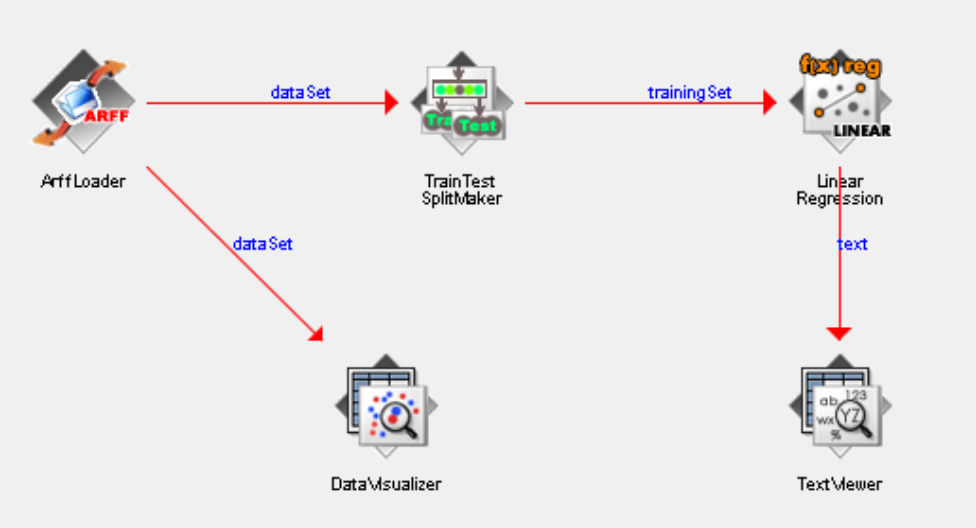
- KnowledgeFlow pozwala na zdefiniowanie procesu przetwarzania danych
- Komponenty realizujące poszczególne czynności można konfigurować, np. podać nazwę pliku do odczytu, parametry operacji lub algorytmu
- Edytor zapewnia poprawność połączeń
- Uruchom z poziomu okna startowego



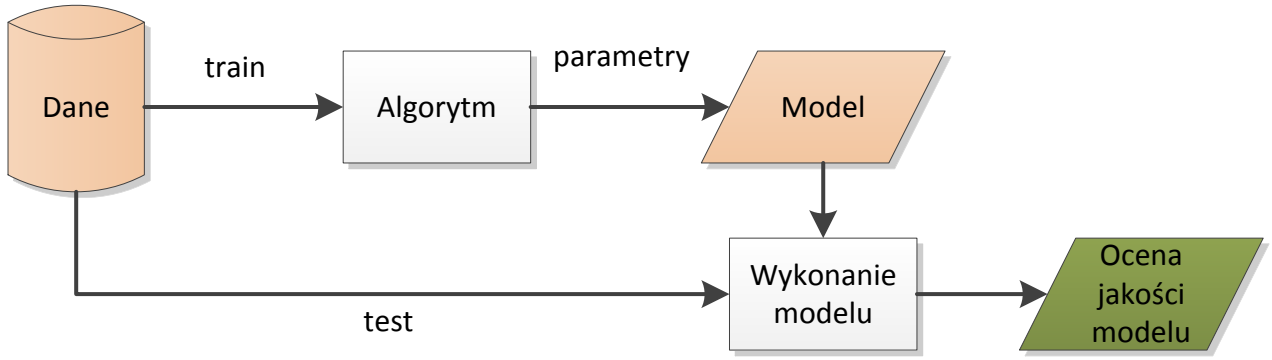
2.1 KnowledgeFlow – prosty proces

- Spróbuj zdefiniować pokazany proces
- Skonfiguruj ArffLoader tak, aby czytał zbiór danych **xy-009.arff**
- Kliknij prawym klawiszem na źródłowym komponencie, aby utworzyć połączenia
- Po uruchomieniu procesu ▶ możesz kliknąć na komponent i wybrać prawym klawiszem dostępne polecenia





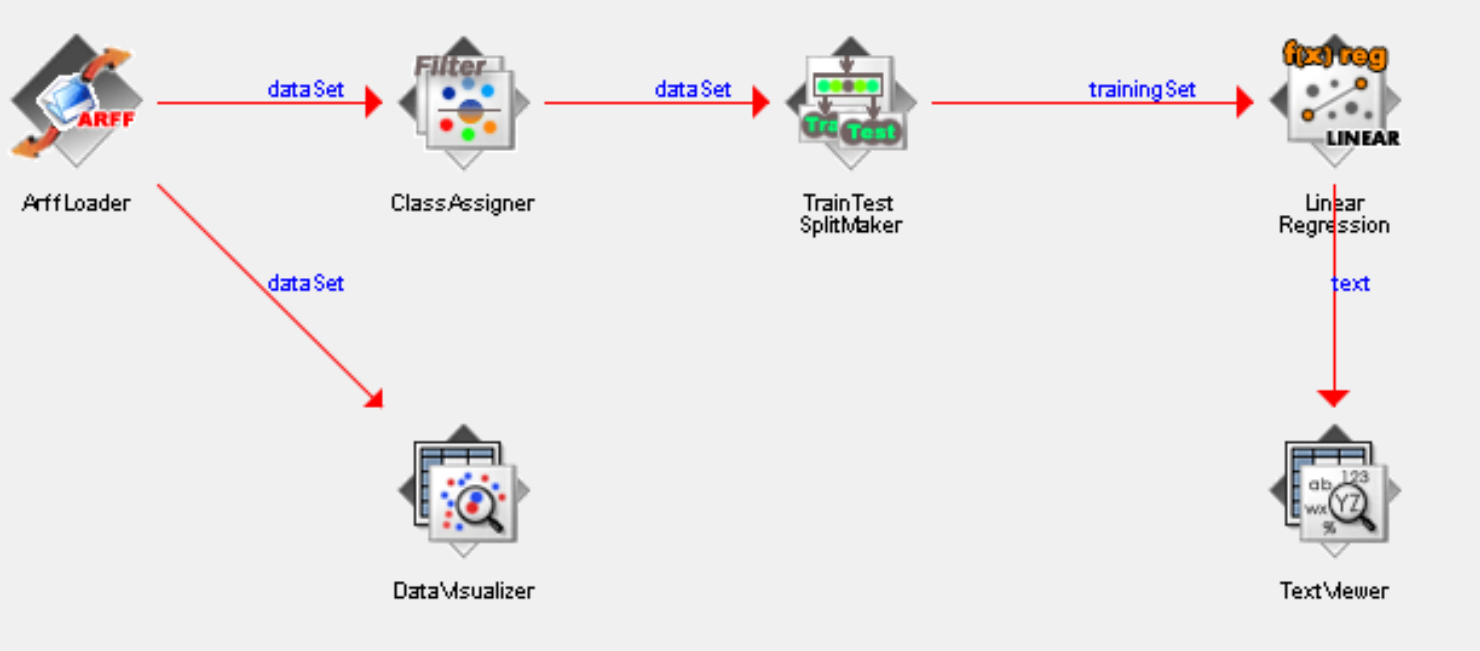
KnowledgeFlow - po co TrainTestSplitMaker?



- Wyznaczanie modelu nie powinno ograniczać się wyłącznie do uczenia. Cały proces obejmuje także jego ocenę przeprowadzaną z użyciem innych danych niż użyte do uczenia.
- Edytor KnowledgeFlow nie pozwala na dostarczenie całego zbioru na wejście algorytmu uczenia (musimy dostarczyć trainingSet)

KnowledgeFlow – poprawiony proces

- Po uruchomieniu pojawiają się błędy? Algorytm nie wie, który atrybut jest wyjściowy.
- Zmień workflow



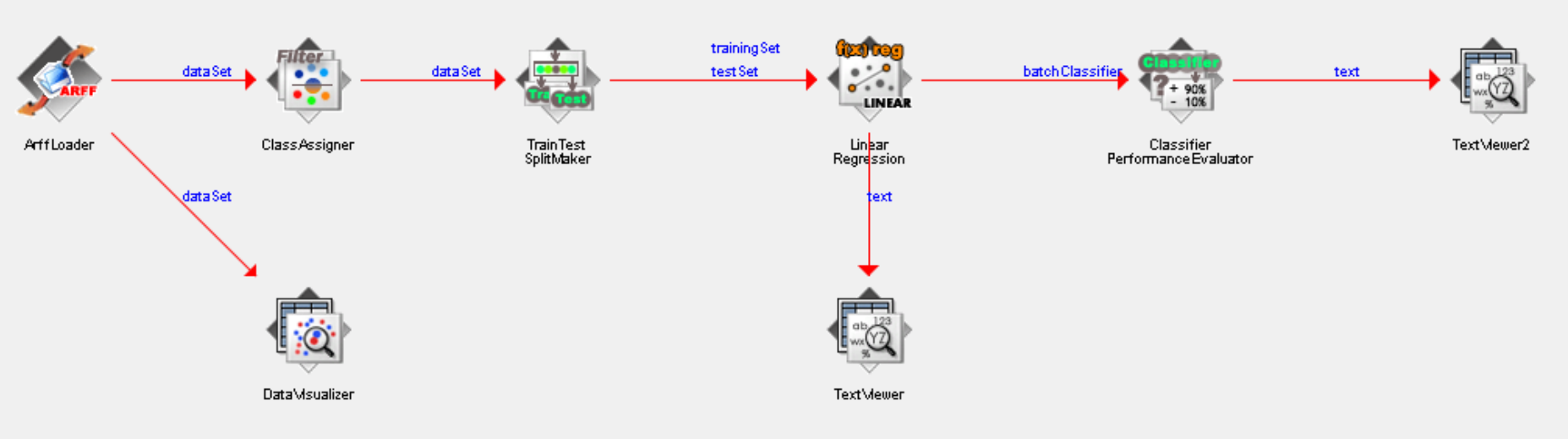
✓ Oczekiwany wynik:

Linear Regression Model

Y = -1.6615 * X + 13.2711

2.2 KnowledgeFlow – dodajemy ocenę

- Rozszerz workflow
- Nie zapomnij dołączyć przepływu testSet



KnowledgeFlow – wyniki

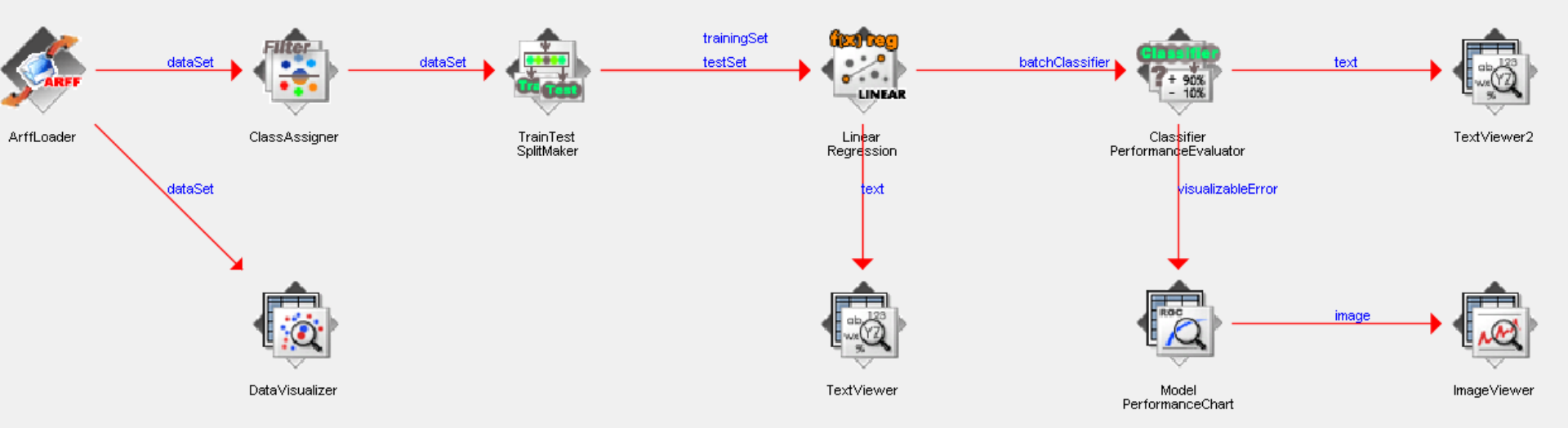
Text

```
=== Evaluation result ===

Scheme: LinearRegression
Options: -S 0 -R 1.0E-8 -num-decimal-places 4
Relation: XY-(ellipse)-weka.filters.unsupervised.attribute.ClassAssigner-Clast


Correlation coefficient                  0.9766
Mean absolute error                      0.826
Root mean squared error                  0.9689
Relative absolute error                 21.5026 %
Root relative squared error             21.5537 %
Total Number of Instances              340
```

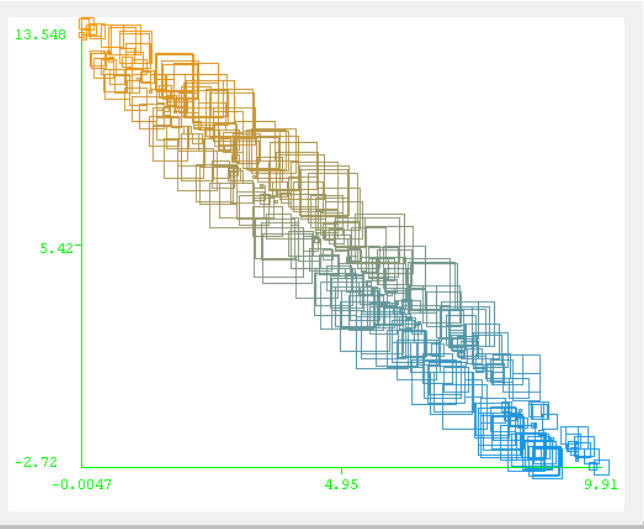
- Sprawdź, co oznaczają poszczególne wskaźniki: http://stats.stackexchange.com/questions/131267/how-to-interpret-error-measures-in-weka-output
- Dlaczego Total Number of Instances = 340 ?

2.3 KnowledgeFlow – dodatkowa wizualizacja

- Dodaj kolejne elementy i uruchom.

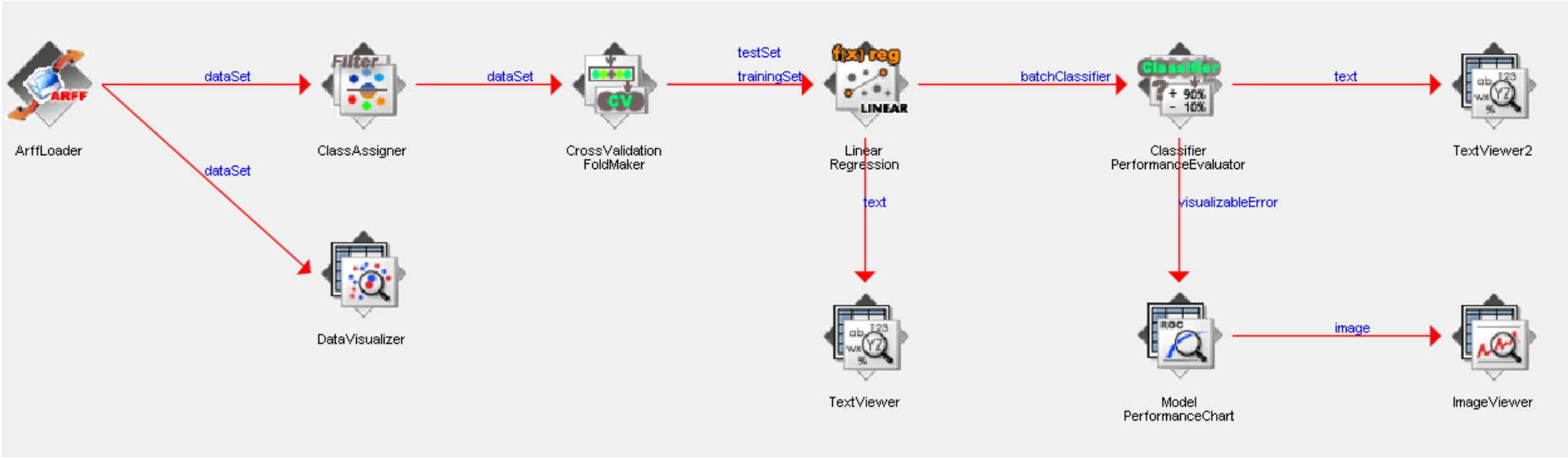


- Wyświetl wynik
- Dlaczego prostokąty oznaczające błędy układają się w elipsę?
- Skąd biały pasek pośrodku?



2.4 KnowledgeFlow – zmień sposób ewaluacji

- Użyj elementu CrossValidationFoldMaker



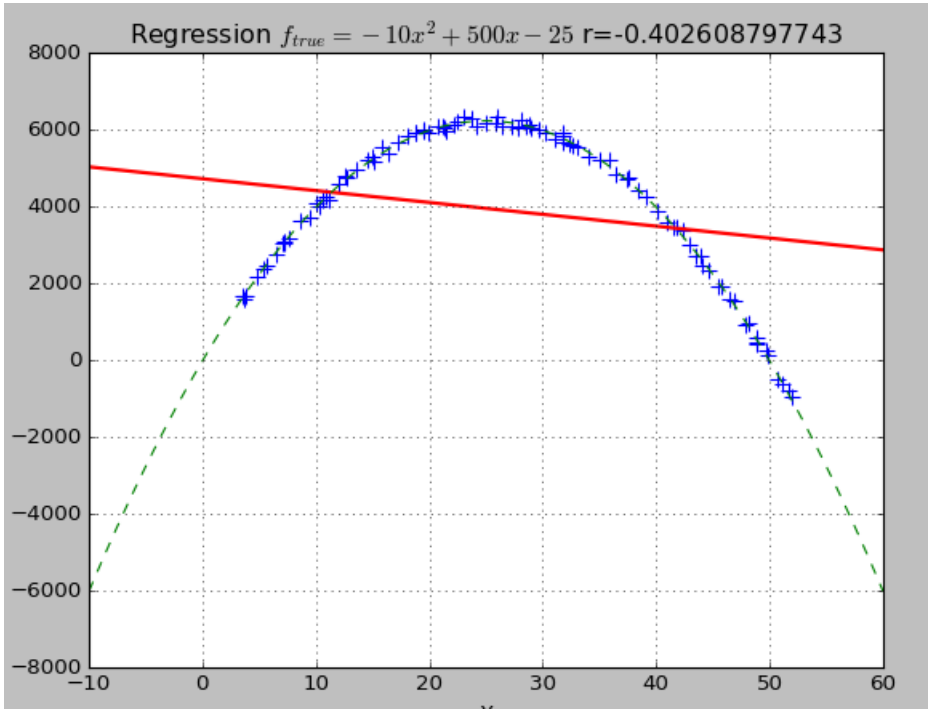
- CV (cross validation) polega na podziale zbioru na k (k=10) podzbiorów
- Uczenie i testowanie zachodzi w k iteracjach
  - k-1 części (folds) użyte do uczenia
  - 1 część użyta do testowania
- W sumie: do uczenia i walidacji użyte są wszystkie instancje
- Obliczane są średnie wyniki

Text

```
=== Evaluation result ===

Scheme: LinearRegression
Options: -S 0 -R 1.0E-8 -num-decimal-places 4
Relation: XY-(ellipse)-weka.filters.unsupervised.attribute.ClassAssigner-Clast


Correlation coefficient                  0.975
Mean absolute error                      0.825
Root mean squared error                  0.9704
Relative absolute error                 22.1431 %
Root relative squared error             22.1937 %
Total Number of Instances             1000
```

2.5 Rozszerzanie zbioru cech

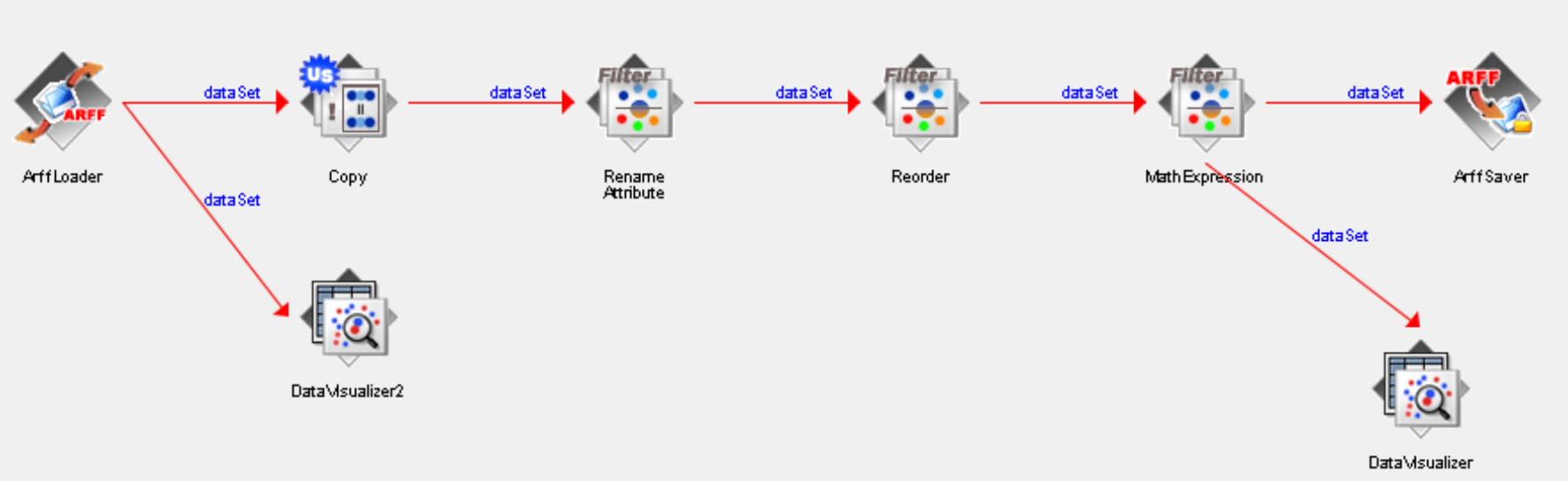
- Przetwarzany plik: **xy-004.arff**
- Uruchom zadanie regresji i odczytaj równanie krzywej
- Wyświetl dane + przebieg funkcji za pomocą odpowiednio zmodyfikowanego skryptu w języku Python
- Oczekiwany rezultat:



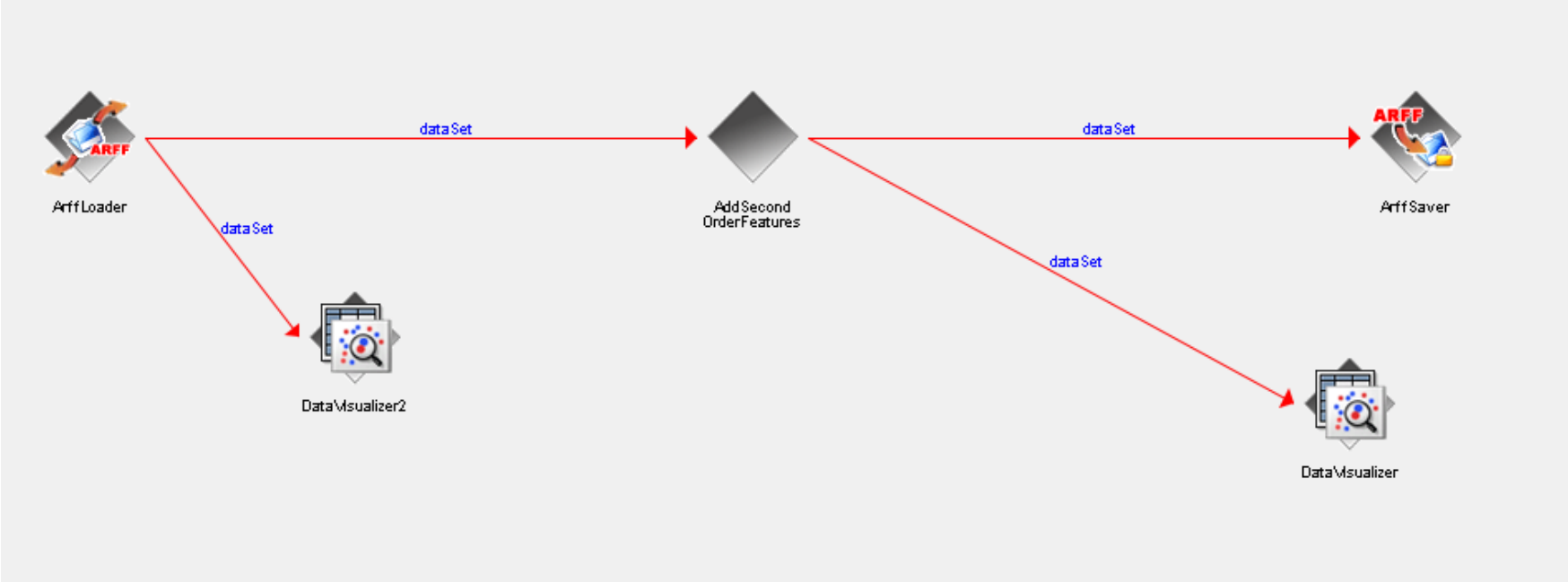
Rozszerzanie zbioru cech

Zimplementuj poniższy proces konfigurując odpowiednio kolejne operacje:

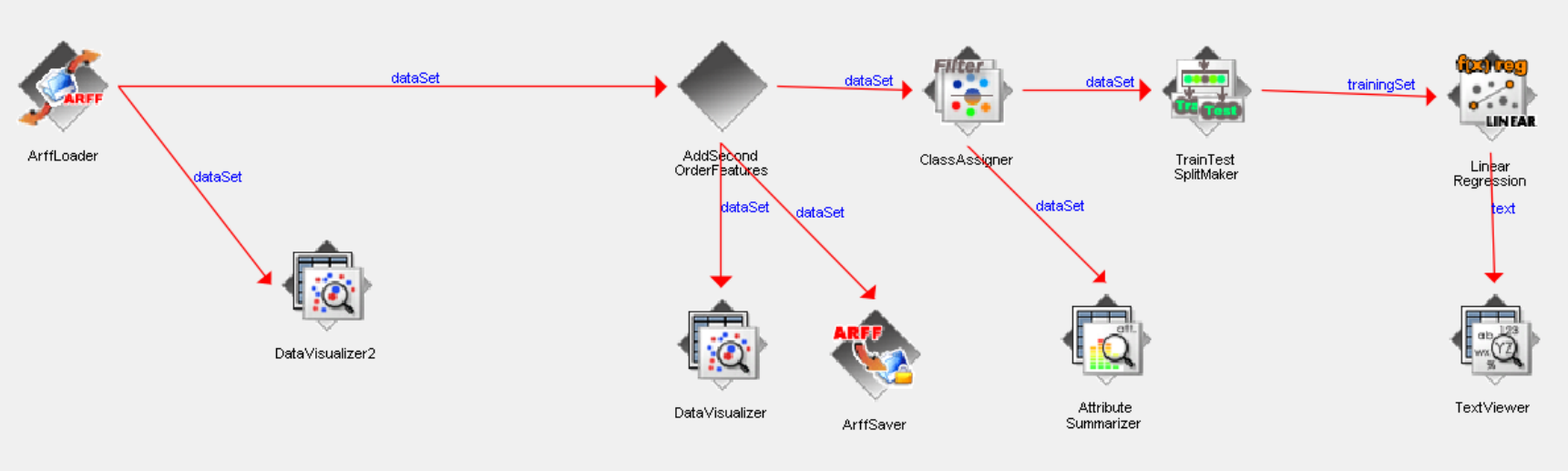
- Skopiowanie atrybutu X
- Zmiana nazwy na X2
- Zmiana porządku tak, aby Y był ostatni
- Podniesienie X2 do kwadartu
- Zapis do pliku (aby sprawdzić poprawność).
  Wybierz np. nazwę **xy-004-01.arff**



Zgrupuj fragment procesu

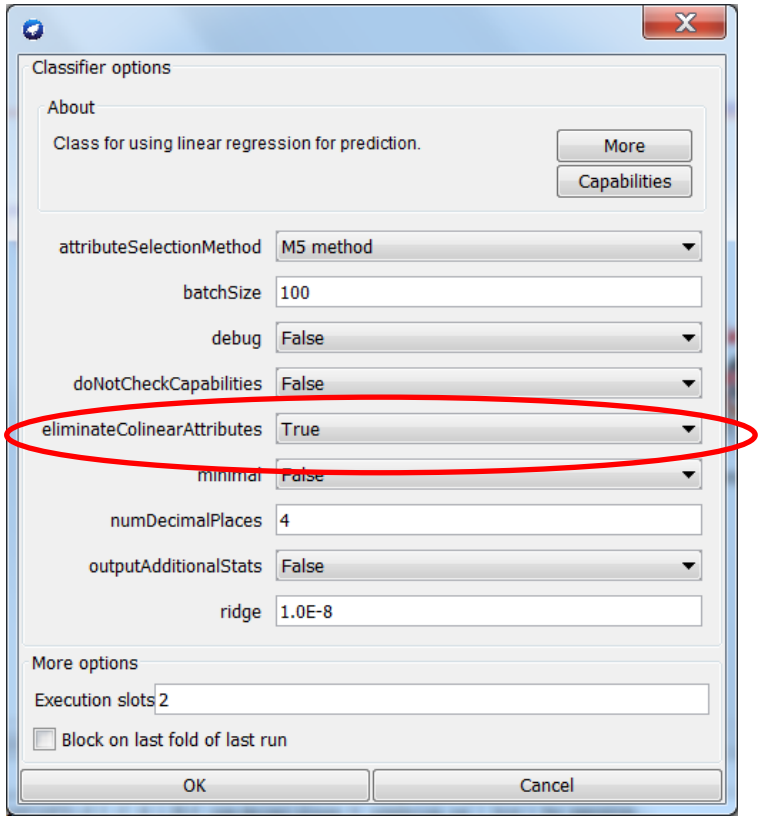


Dodaj komponenty odpowiedzialne za przeprowadzenie regresji



Uruchom i sprawdź wynik.
Jeśli ma on postać X2 = a*X → Źle
Jeśli ma postać Y = stała → Źle, ale to kwestia konfiguracji algorytmu

Konfiguracja algorytmu regresji



- Zmień parametr na false
- Spójrz na wykres X/X2 W tym zakresie atrybuty są bliskie liniowym.

✓ Równanie regresji powinno mieć współczynniki bliskie rzeczywistej funkcji

✓ Sprawdź błędy walidacji. Błąd względny powinien być kilkuprocentowy

✓ Użyj kodu Pythona do narysowania danych oraz zależności x->y

2.6 Wprowadź cechy 3 stopnia

- Zbuduj KnowledgeFlow dla pliku **XY-005.arff**
- Wprowadź cechę 3 stopnia (x^3)
- Znajdź współczynniki regresji
- Wyświetl dane i wyznaczoną krzywą posługując się skryptem w języku Python

2.7 Rozwiązanie w języku Python

- Przetwarzamy **xy-004.arff:** zapisz w KnowledgeFlow plik z dodatkowym atrybutem X2, np. jako xy-004a.arff
- Skopiuj zawartość pliku do skryptu Pythona lub wczytaj z pliku

  x, x2, y = np.loadtxt(inp, delimiter=',', usecols=(0, 1, 2), unpack=True,skiprows=?)
- Utwórz dwuwymiarową macierz cech:

  features = np.stack((x,x2),axis=-1)
- Dopasuj krzywą

  regr = linear_model.LinearRegression()
  regr.fit(features, y)
- Wyświetl wynik