# Applying Machine Learning Algorithms on Synchrotron X-Ray Pair Distribution Data

Theo Dimitrasopoulos

October 6, 2016

# Chapter 1

# Introduction

## 1.1 Introduction

Machine learning techniques have been widely developed and implemented across a variety of disciplines to provide insights and extract patterns from large datasets. This tool is especially useful in predicting future behaviors and creating coherent models that address problems that arise from long experiments that have unrealistic timeframes and cannot be performed due to such time limitations. By applying prediction algorithms on X-Ray Pair Distribution data we will be able to predict at what times certain peaks of interest arise and with enough accuracy be able to predict values for longer and longer timelines.

## 1.2 Research Problem

Claire W. White et al. in their "In situ synchrotron X-ray pair distribution function analysis of the early stages of gel formation in metakaolin-based geopolymers" outline the process of investigation of the evolution of metakaolin-based geopolymers through in-situ X-ray pair distribution function analysis. The atom-atom correlations that were identified show peaks occurring for a variety of samples for different curing stages. However, longer curing times that are especially relevant in construction are not possible because the experiment running time is unrealistic and also the distance between atoms $r$ in the PDF's generated does not continue for values past 8-11 Å.

## 1.3 Research Objectives

With the aid of statistical inference and machine learning algorithms we are hoping to predict values for later curing stages and predict atom-atom correlations and peaks that were not observed because the experiment wasn't allowed to run for a longer period of time.

# Chapter 2

# Literature Review

Claire E. White et al.'s "In situ synchrotron X-ray pair distribution function analysis of the early stages of gel formation in metakaolin-based geopolymers" and Provis et al's "X-ray microtomography shows pore structure and tortuosity in alkali-activated binders" were studied in order to get a basic idea of the experiment design and the nature of the data being generated. The final graphs run the atomic distance $r(\mathring{A})$ against the strength of the atom-atom interactions $G(r)(\mathring{A}^{-2})$ and presents peaks for individual pairs at set distances. No research in the past has been involved in data analysis using Machine Learning algorithms.

# Chapter 3

# Previous Research

The main statistical algorithm used to infer information from the preexisting data is Isotonic Regression. The regression function used is the following for a simply ordered case,

$$\min_{g \in A} \sum_{i=1}^{n} w_i (g(x_i) - f(x_i))^2$$

where the isotonic estimator $g^*$ extracted from the data minimizes the weighted least-squares problem. In a more genreral case, we solve the following minimization problem,

$$\min \sum_{i=1}^{n} w_i (x_i - a_i)^2 \text{ subject to } x_i \leq x_j \forall (i,j) \in E$$

# Chapter 4

# Data Presentation and Analysis

Attached is the main algorithm used to extract the peaks from the dataset

```
import pandas as pd
import numpy as np
for i in range (1,61):
    iterable = pd.read_csv('/PATHTOFILE', sep=',',names = ['r','G'])
    area1 = iterable[3000:5000]
    print np.max(area1.G)

final = pd.read_csv('/PATHTOFILE', sep=',',names = ['r','G'])
area2 = final[3000:5000]
print np.max(area2.G)

longterm = pd.read_csv('/PATHTOFILE', sep=',',names = ['r','G'])
area3 = longterm[3000:5000]
print np.max(area3.G)
```

And the main sampling algorithm that extracts the regressors and minimizes the least-squares function,

```
"""
================================================================
G(r) Isotonic Regression for High-Alkali, H-activated Metakaolin
================================================================
"""

print(__doc__)

import numpy as np
import matplotlib.pyplot as plt
```

```
import pandas as pd
from matplotlib.collections import LineCollection

from sklearn.linear_model import LinearRegression
from sklearn.isotonic import IsotonicRegression
from sklearn.utils import check_random_state

main = pd.read_csv('PATHTOFILE', sep=',',names = ['Time','G'])
mainx_data = main.Time[1:60]
mainx_target = main.G[1:60]

###############################################################
# Fit Isotonic Regression model
###############################################################

ir = IsotonicRegression()
lr = LinearRegression()

y_ = ir.fit_transform(mainx_data, mainx_target)
predictions = ir.predict([10])
print predictions
print ir.score(mainx_data, mainx_target)
#print("RSS: \%.2f"
#      \% np.mean((ir.predict(mainx_target) - mainy_target) ** 2))

###############################################################
# Plot result
###############################################################

fig = plt.figure()
plt.plot(mainx_data, mainx_target, 'r.', markersize=12)
plt.plot(mainx_data, y_, 'g.-', markersize=12)
plt.legend(('Data', 'Isotonic Fit', 'Linear Fit'), loc='lower right')
plt.title('Isotonic regression')
plt.show()
```

As can be seen in the graphs, the farthest the algorithm could predict values within a certain accuracy was 10Å.

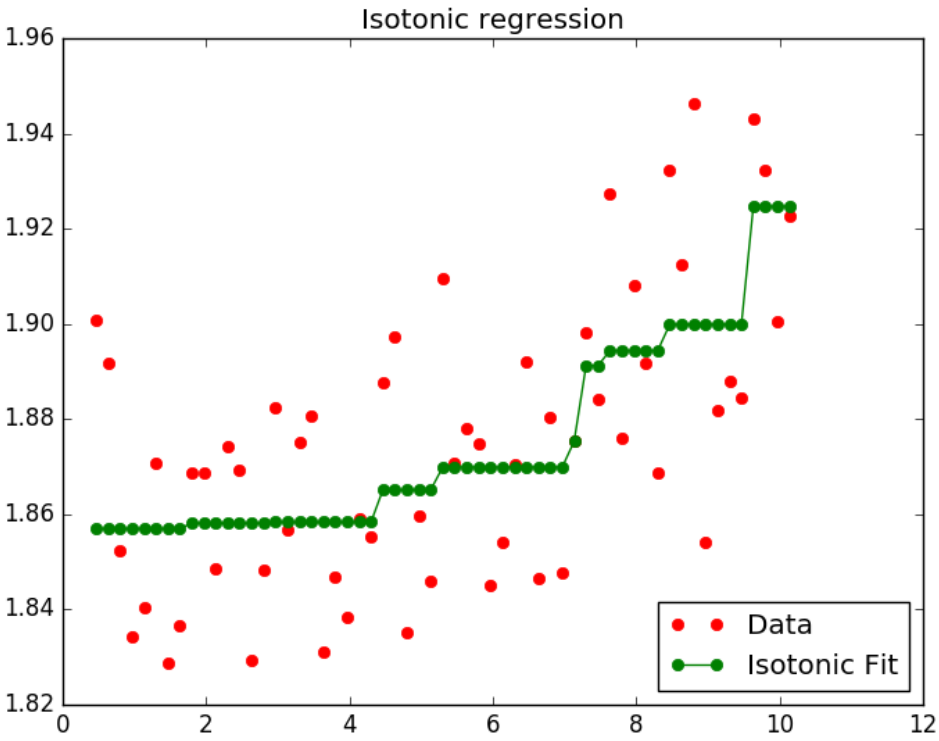


Figure 4.1: Isotonic Curve for the High-Alkali H-Activated Metakaolin

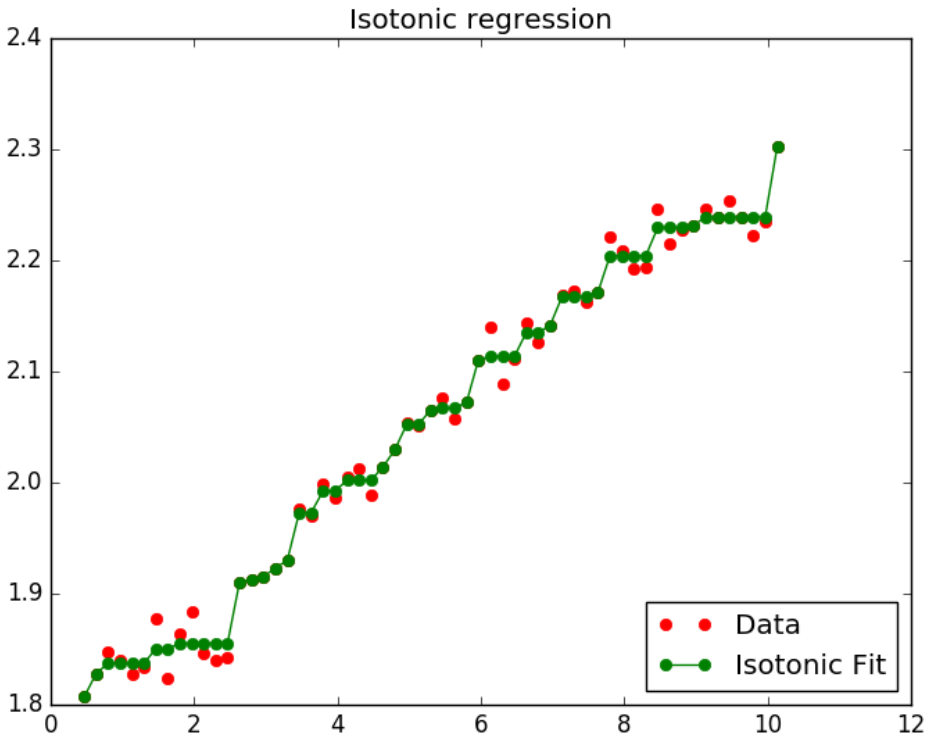


Figure 4.2: Isotonic Curve for the High-Alkali S-Activated Metakaolin

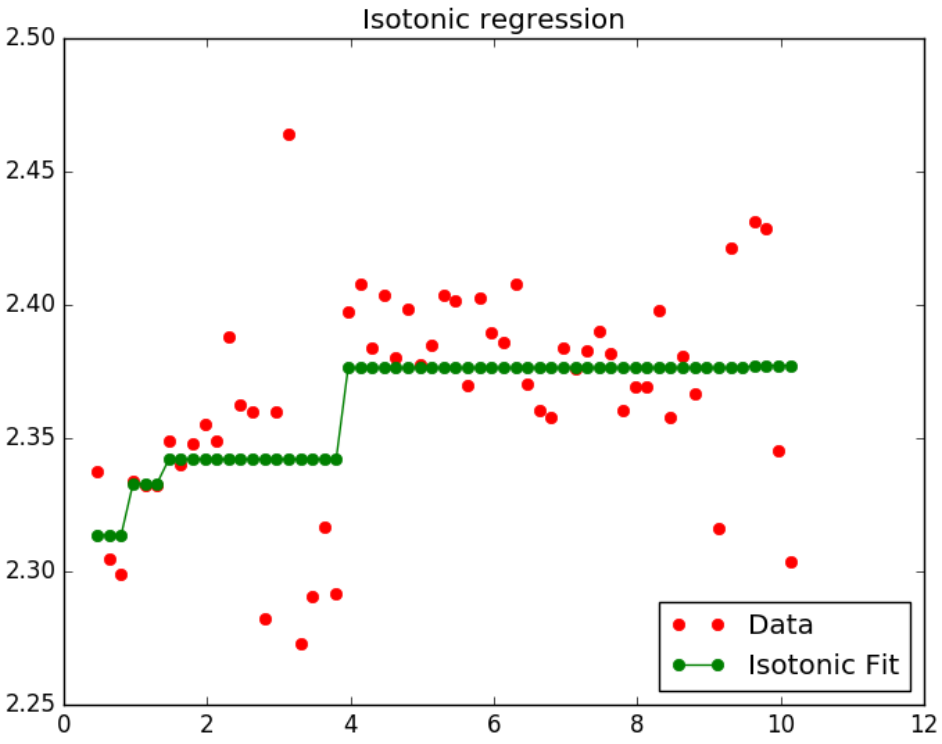


Figure 4.3: Isotonic Curve for the Low-Alkali H-Activated Metakaolin

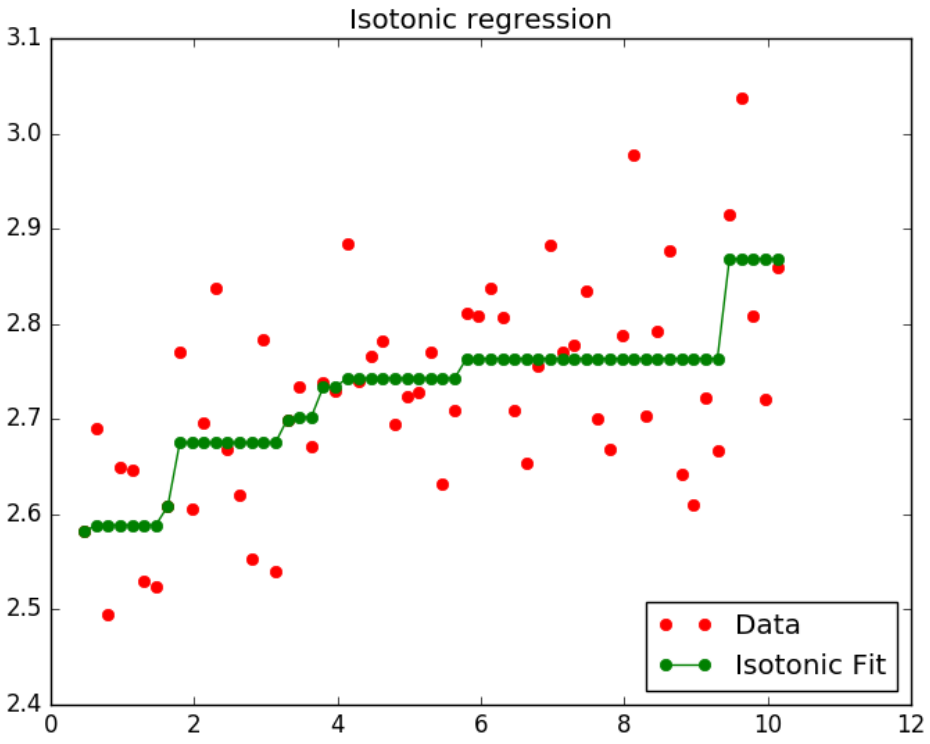


Figure 4.4: Isotonic Curve for the Low-Alkali S-Activated Metakaolin

# Chapter 5

# Summary and Conclusions

The main issue that arose during the programming analysis was the R-Values are rather low, pointing towards an inaccuracy in the predicted values generated. However, whenever an already know value for $G(r)$ or the time was used in the algorithm, the error was as low as 4% in some cases. This error is pointing towards a small dataset, which could explain the inaccuracy past a value close to the ones that we already know. A more large dataset could minimize the error and generate the desired predicted values with more accuracy while keeping the same algorithm to analyze the data.