Analysis of alloy steel composition-property relationship using machine learning

Raymond Wang[1]

[1] *Department of Materials Science and Engineering, Northwestern University, Evanston, Illinois 60208, USA*

This challenge project obtains digital alloy steel data from online resources and then converts them into machine-readable format. Data analysis using correlation heatmap shows degree of correlation within alloy steel composition and properties. Several strongly correlated quantities (e.g. hardness and tensile strength) are identified. Machine learning algorithm performance on predicting composition-property relationship is benchmarked using grid search. XGBoost outperforms the other methods in this case. Further analysis shows predicting thermal conductivity from chemical composition using XGBoost has satisfying accuracy and the model performance can be improved by having more training data. Our findings suggest that machine learning methods could provide more insights of alloy steel composition-property relationship than using human intuition or experience.

I. WORKFLOW

This challenge project mainly consists of five sections, i.e. data acquisition, featurization, human interpretation, machine learning model training, and results analysis. The flowchart and specific outcomes in each step are shown in Figure 1. The rest of this report will span these five sections into details of implementation and obtained results. Data is available at the author's GitHub, the code can be found in supporting information (SI).

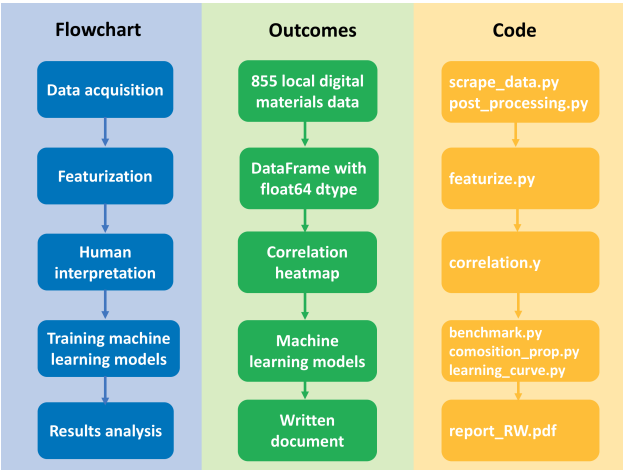

FIG. 1. Flowchart of the challenge project.

II. DATA ACQUISITION

The challenge project starts from acquiring materials data from online resources. Automated scraping code is developed by the author in Python, which automatically collects materials information that meets the searching criteria and saves to local CSV files.d by the author.

Materials data is collected from MatWeb, where alloy steels containing Manganese, Chromium, and Nickel are set as the target materials for scraping. Each entry of material data spans four columns and multiple rows, as shown in Table I. Data under 'English' has different unit from 'Metric', this project only processes data with metric unit.

TABLE I. Format of online resources.

Property	Metric	English	Comments
Elongation at break	55%	55%	some comments
...	...	...	...

Both physical properties (e.g. Bulk modulus, thermal conductivity) and chemical compositions (e.g. weight of Mn, Ni, Cr) in different units, as well as the potentially helpful comments, are saved to local files with the same format. Since there is no data-editing related in this step, the local data is guaranteed to be the same as its online version.

One of the technical challenges during data acquisition is IP-blocking from the server. MatWeb will block the IP (possibly permanently) once it detects over access within a short period of time. From the author's experience, one IP address gives access to 100∼200 materials information per day. One of the advantage of the developed code is its high portability. The only two dependencies are `selenium` and `pandas` packages, which can be installed easily. In order to accelerate the project, the author used six machines with Unix-like systems for development and production. Eventually the program extracted 855 alloy steels that meet the criteria. The code (`scrape.py`) is available in SI.

Simple post-processing is performed right after all the data has been saved locally. This step modifies the file names and contents containing non-utf-8 encoding and fixes unwanted line breaks. The process file is still string-based, relevant code is available is SI (`post_processing.py`).

III. FEATURIZATION

The data collected from the last step is still string-based, e.g. "97%" is interpreted as a combination of characters instead of a floating point number. Therefore, it is necessary to convert these strings to machine-readable form before any further data analysis.

Since there are multiple materials properties and not all of them are available for each materials, dataset with missing values will be dropped. In order to keep a relatively large number of training set, we only converted physical properties with more than 100 available measured data points. These physical variables are: density, hardness (Vickers), thermal conductivity, specific heat capacity, CTE-linear, electrical resistivity, elongation at break, bulk modulus, modulus of elasticity, shear modulus, poisson's ratio, tensile strength at yield, and tensile strength at ultimate. Ten element types including Fe, Mn, Cr, Ni, Mo, Cu, C, S, Si, P are considered.

Floating point numbers are extracted from string data and converted to `pandas.DataFrame` format. The converted dataset is shown in Table II. Non-available data points are converted to `nan` instead of 'N/A'. The '%' and other units are dropped, only the floating point numbers are extracted. Since data will be standardized before numerical processing, the only thing to make sure is that all data in the same column share the same unit/percentage sign. The code is available in SI (`featurize.py`).

TABLE II. Format of data after featurization.

Fe	Mn	Cr	···	Hardness	Bulk modulus	···
97.16	0.88	0.5	···	220	`nan`	···
95.43	0.85	0.0	···	`nan`	160	···
···	···	···	···	···	···	···

IV. DATA ANALYSIS WITH HUMAN INTELLIGENCE

The size of the dataset after featurization is 726×23 (with 726 instances and 23 features). It is impractical for humans (at least for the author) to directly learn patterns from such large amount of data. Based on basic statistical knowledge, the author decides to start from learning correlation patten of these variables. The instances with `nan` entries are dropped from the dataset, and eventually 254 materials are used to generate the heatmap, which is shown in Figure 2. The code can be found in SI (`correlation.py`)

Some of the discoveries from the heatmap include:

i. Carbon increases electrical resistivity but decreases shear modulus

ii. Sulfur increases specific heat capacity

iii. Phosphorus increases electrical resistivity

iv. Chromium decreases density

v. Molybdenum and copper strongly increases specific heat capacity and electrical resistivity

vi. Nickel increases thermal conductivity

vii. Hardness is positively correlated to tensile strength and negatively correlated to elongation at break

viii. Electrical resistivity is positively related to specific heat capacity

Some of the findings agree well with the way elements contribute to alloy steel properties as reported online, e.g. carbon decreases ductility of steel, possibly could lead to a small shear modulus. However, the information from correlation analysis is more qualitative than quantitative. If we want more quantitative descriptions of the materials composition-property relationship, more sophisticated methods are needed. In the next section we discuss alloy steel data analysis using machine learning models.

V. DATA ANALYSIS WITH ARTIFICIAL INTELLIGENCE

A. benchmark machine learning algorithm performance on dataset

The author perform systematic benchmark of different machine learning algorithm performance on various physical property predictions.

Each training takes one physical property as the target, while treats the rest 12 properties plus aforementioned 10 element types as input features. Dataset instances with `nan` are dropped from the dataset. Numerical data (both X and y) are standardized as implemented in `StandardScaler` in `sklearn`.

10 machine learning algorithms are benchmarked here, including:

1. Linear Regression
2. Least-angle regression with Lasso
3. Kernel Ridge
4. Linear SVR
5. SGD Regression
6. MLP Regressor
7. AdaBoost Regression
8. Random Forest Regression
9. Gradient Boosting Regression
10. Extremen Gradient Boosting

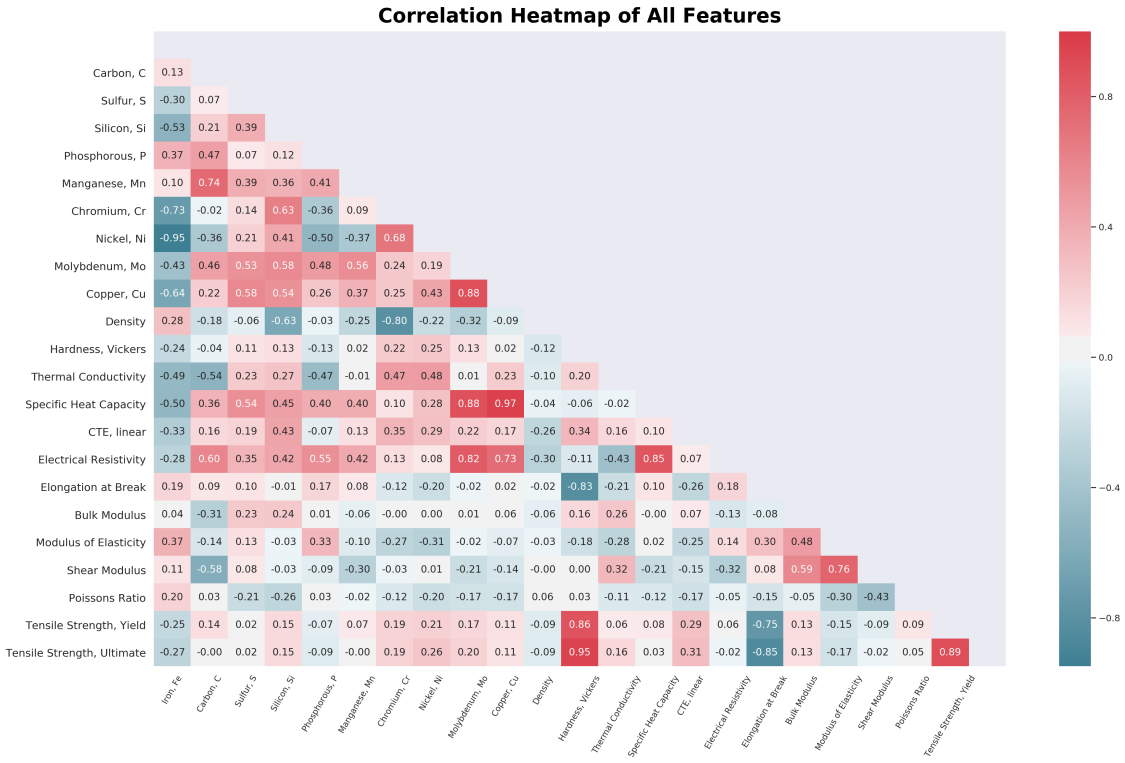

FIG. 2. Correlation heatmap of chemical composition and physical properties.

The dataset is divided into training (75%) and testing (25%) parts. Grid search method is used for hyper-parameter tuning. The set of hyper-parameters are shown in Table III . The optimal setting of the parameters is determined based on 5-fold cross-validation performed on training data only. R2 score is used as error metric. Relevant code can be found in SI (`benchmark.py`). Results are shown in Figure 3.

The y-axis is the calculated root mean square error (RMSE) value divided by the difference in the target data. This normalization step is essential so as to directly compare the performance of different machine learning algorithms in predicting quantities at various scales. The `sklearn` built-in performance score is not used here.

Interestingly, some trends can be identified across different algorithms as well as physical properties. In general, linear regression, Lasso, linear SVR and SGD algorithms lead to larger variance in normalized RMSE, while the rest have relatively better performance. XGBoost algorithm exhibits the best performance among the 10 benchmarked methods.

The physical properties are divided into two panels due to different scales of RMSE. We find it hard to have a good prediction in hardness and tensile strength, but this is no coincidence. It is exciting to notice that from the correlation heatmap analysis, it is clear that hardness is strongly and positively correlated to tensile strengths. This may suggest that these quantities are related to some other factors not considered here, such as processing and microscopic structures. And we need to include the other important features in order to build better machine learning models.

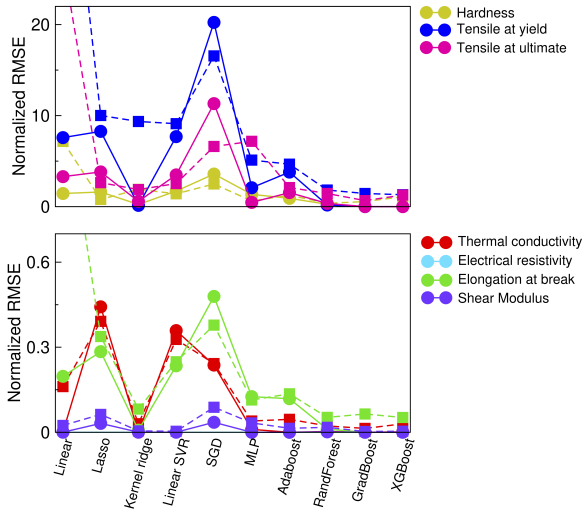

FIG. 3. Normalized root mean square error (RMSE) of different machine learning algorithms in predicting various physical properties. Training errors are shown in circles and solid lines, test errors are in squares and dashed lines.

TABLE III. Hyper-parameters used in different machine learning algorithms for grid search.

Algorithm	Parameter	Values
Linear Regression	default	default
Lasso	'alpha'	{ 1E-4, 0.001, 0.01, 0.1, 1 }
Kernel Ridge	'kernel'	{'linear', 'poly', 'rbf', 'sigmoid' }
	'alpha'	{ 1E-4, 1E-2, 0.1, 1 }
	'gamma'	{ 0.01, 0.1, 1, 10 }
Linear SVR	'C'	{ 1E-6, 1E-4, 0.1, 1 }
	'loss'	{ 'epsilon_insensitive', 'squared_epsilon_insensitive'}
SGD	'alpha'	{1E-6, 1E-4, 0.01, 1 }
	'penalty'	{'l2', 'l1', 'elasticnet'}
MLP	'activation'	{'logistic', 'tanh', 'relu'}
	'solver'	{'lbfgs', 'adam', 'sgd'}
	'learning_rate'	{'constant', 'invscaling', 'adaptive'}
Adaboost	'n_estimators'	{10, 100, 1000}
	'learning_rate'	{0.01, 0.1, 1, 10}
RandForest	'n_estimators'	{10, 100, 1000}
	'min_weight_fraction_leaf'	{0.0, 0.25, 0.5}
	'max_features'	{'sqrt', 'log2', None}
GradBoost	'n_estimators'	{10, 100, 1000}
	'min_weight_fraction_leaf'	{0.0, 0.25, 0.5}
	'max_feature'	{'sqrt', 'log2', None}
XGBoost	'n_estimators'	{10, 50, 100, 250, 500, 1000}
	'learning_rate'	{1E-4, 0.01, 0.05, 0.1, 0.2}
	'gamma'	{0, 0.1, 0.2, 0.3, 0.4}
	'max_depth'	{6}
	'subsample'	{0.5, 0.75, 1}

B. predict composition-property relationship

From the previous section we choose to use Extreme Gradient Boosting algorithm for further analysis on alloy steel composition-relationship problems.

VI. CONCLUSIONS

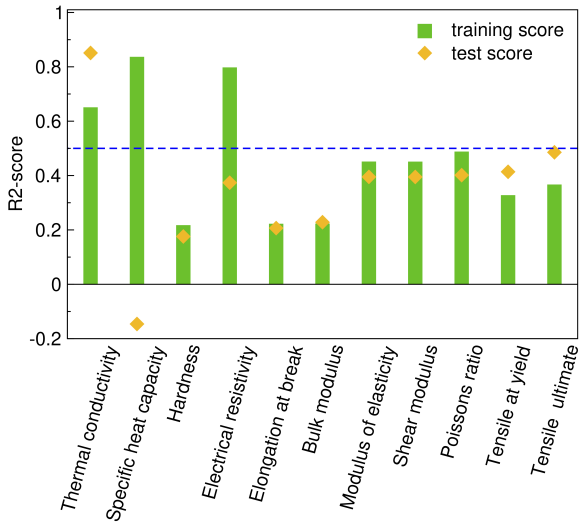

FIG. 4. ADD CAPTION

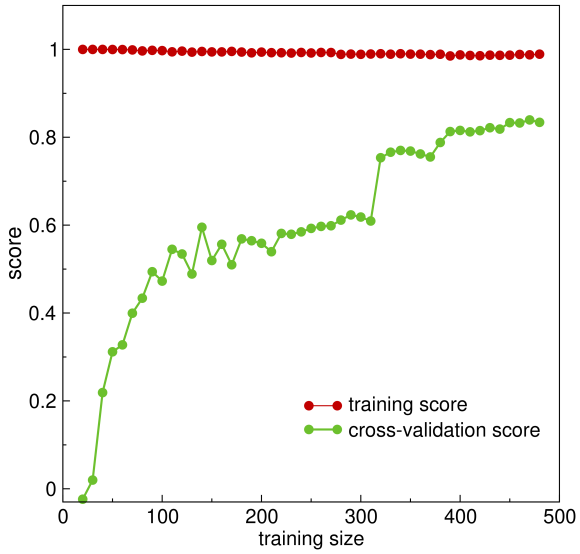

FIG. 5. Learning curve of predicting thermal conductivity from chemical composition using XGBoost algorithm.