

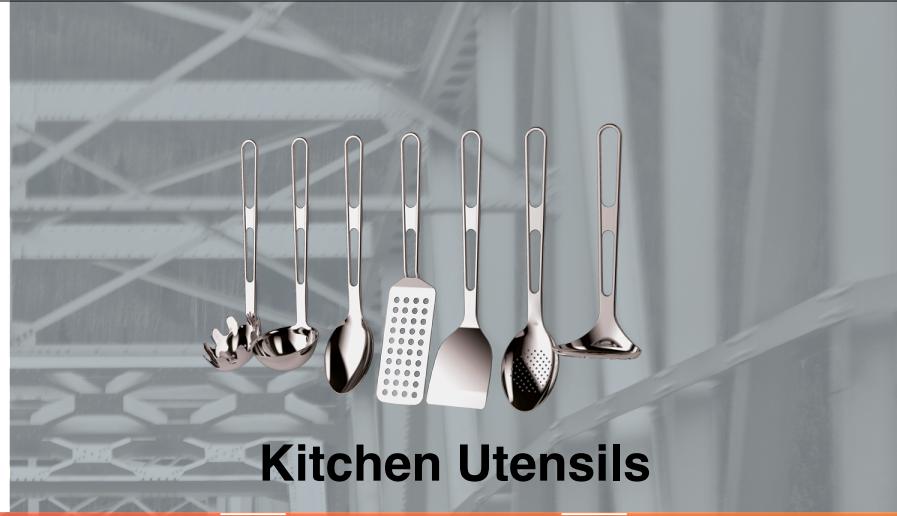
# Steel Grade Classification

James Fan | Metis SF | 8/7/19





**Automotive Parts**



**Kitchen Utensils**



**Disney Concert Hall**



Companies over specify more expensive materials, when more cost efficient alternatives are available.



# Steel Specification

## 2.1 Chemical Composition of Carbon Steels for General Use (Continued)

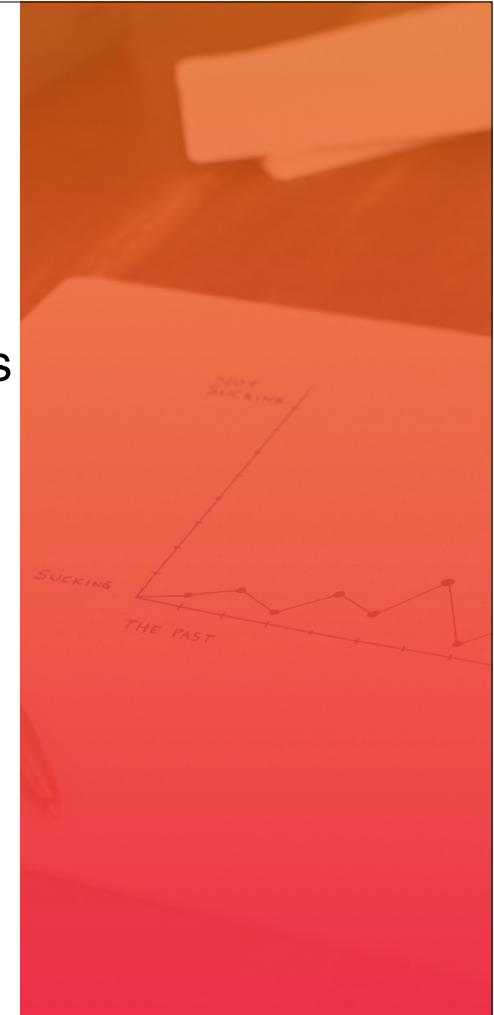
Standard Designation	Grade, Class, Type, Symbol or Name	Steel Number	UNS Number	Weight, %, max, Unless Otherwise Specified								
				C	Mn	Si	P	S	Cr	Ni	Mo	Others
ASTM A 29/A 29M-03	1049	---	G10490	0.46-0.53	0.60-0.90	---	0.040	0.050	---	---	---	---
	1050	---	G10500	0.48-0.55	0.60-0.90	---	0.040	0.050	---	---	---	---
	1053	---	G10530	0.48-0.55	0.70-1.00	---	0.040	0.050	---	---	---	---
ASTM A 108-03	1050	---	G10500	0.48-0.55	0.60-0.90	---	0.040	0.050	---	---	---	---
	1049	---	G10490	0.46-0.53	0.60-0.90	---	0.040	0.050	---	---	---	---
	1050	---	G10500	0.48-0.55	0.60-0.90	---	0.040	0.050	---	---	---	---
ASTM A 576-90b (2000)	1053	---	G10530	0.48-0.55	0.70-1.00	---	0.040	0.050	---	---	---	---
	1049	---	G10490	0.46-0.53	0.60-0.90	---	0.030	0.050	---	---	---	---
	1050	---	G10500	0.48-0.55	0.60-0.90	---	0.030	0.050	---	---	---	---
SAE J403 NOV01	1053	---	G10530	0.48-0.55	0.70-1.00	---	0.030	0.050	---	---	---	---
	1049	---	G10490	0.46-0.53	0.60-0.90	---	0.030	0.050	---	---	---	---
	1050	---	G10500	0.48-0.55	0.60-0.90	---	0.030	0.050	---	---	---	---
JIS G 4051:1979	S 50 C	---	---	0.47-0.53	0.60-0.70	0.15-0.25	0.030	0.035	0.20	0.20	---	Cu 0.30; Ni+Cr 0.35
EN 10016-2:1995	C50D	1.0586	---	0.48-0.53	0.50-0.80	0.10-0.30	0.035	0.035	0.15	0.20	0.05	Cu 0.25; Al 0.01
EN 10016-4:1995	C50D2	1.1171	---	0.48-0.53	0.50-0.70	0.10-0.30	0.030	0.025	0.10	0.10	0.03	Cu 0.15; Al 0.01; N 0.007
EN 10083-1:1991	C50E	1.1206	---	0.47-0.55	0.60-0.90	0.40	0.035	0.035	0.40	0.40	0.10	Cr+Mo+Ni 0.63
	C50R	1.1241	---	0.47-0.55	0.60-0.90	0.40	0.035	0.020-0.040	0.40	0.40	0.10	Cr+Mo+Ni 0.63
	C50	1.0540	---	0.47-0.55	0.60-0.90	0.40	0.045	0.045	0.40	0.40	0.10	Cr+Mo+Ni 0.63
ISO 683-1:1987	C 50	---	---	0.47-0.55	0.60-0.90	0.10-0.40	0.045	0.045	---	---	---	---
	C 50 E4	---	---	0.47-0.55	0.60-0.90	0.10-0.40	0.035	0.035	---	---	---	---
	C 50 M2	---	---	0.47-0.55	0.60-0.90	0.10-0.40	0.035	0.020-0.040	---	---	---	---
JIS G 4051:1979	S 53 C	---	---	0.50-0.56	0.60-0.90	0.15-0.35	0.030	0.035	0.20	0.20	---	Cu 0.30; Ni+Cr 0.35
EN 10016-2:1995	C52D	1.0588	---	0.50-0.55	0.50-0.80	0.10-0.30	0.035	0.035	0.15	0.20	0.05	Cu 0.25; Al 0.01
EN 10016-4:1995	C52D2	1.1202	---	0.50-0.54	0.50-0.70	0.10-0.30	0.020	0.025	0.10	0.10	0.03	Cu 0.15; Al 0.01; N 0.007

Source: Handbook of Comparable World Steel Standards 3rd Edition John E Bringas

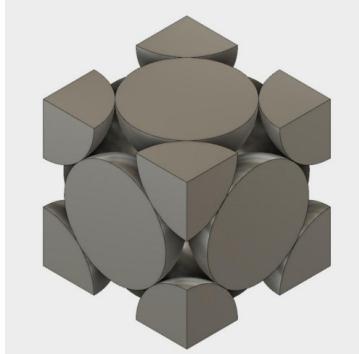
# Problem Statement

- There are thousands of different steel specifications
- Countries have their own standard with different names
- So many options, choosing the right one is hard

**How do you choose the one that works?**



# Steel Type Characteristics



Face Centered Cubic



Body Centered Cubic

Allotrope	Corrosion Resistance	Hardness	Crystal Structure	Relative Cost
Austenite	Excellent	Good	Face Centered Cubic	Mid
Ferrite	Limited	Ok	Body Centered Cubic	Low
Martenite	Poor	Excellent	Body Centered Cubic	Low
Duplex	Excellent	Good	Face Centered Cubic & Body Centered Cubic	High
Super Austenite	Excellent	Good	Face Centered Cubic	Mid-High

# Data



## Target/Label

Austenitic,  
Ferritic,  
Martensitic,  
Duplex,  
Super Austenitic

## Features

Percentage of required  
of each element

1 IA	2 IIA	VIIA																	
H	Be	Boron	Carbon	Nitrogen	Oxygen	Fluorine	Neon												
Hydrogen 1.008	Beryllium 6.94	6.015	12.011	14.007	15.999	18.998403063	20.1797												
Lithium 6.94	Magnesium 24.305	12.011	12.011	14.007	15.999	16.999403063	20.1797												
Sodium 22.9897000	Aluminum 26.981385	13.015	14.014	15.015	16.016	17.017	18.018												
19 K Potassium 39.0983	Ca Calcium 40.078	Sc Scandium 44.955908	Ti Titanium 47.867	V Vanadium 50.944	Cr Chromium 51.9861	Mn Manganese 54.938044	Fe Iron 55.845	Co Cobalt 58.93194	Ni Nickel 58.93194	Cu Copper 63.546	Zn Zinc 65.38	31 Ga Gallium 69.723	Ge Germanium 72.630	As Arsenic 74.971995	Se Selenium 78.971	Br Bromine 79.904	Kr Krypton 83.781		
37 Rb Rubidium 84.94616	38 Sr Strontium 88.80564	39 Y Zirconium 91.224	40 Zr Niobium 92.9063	41 Nb Molybdenum 95.95	42 Mo Technetium (90)	43 Tc Ruthenium 101.07	44 Ru Rhodium 102.90550	45 Rh Rhodium 102.90550	46 Pd Palladium 104.942	47 Ag Silver 107.8662	48 Cd Cadmium 112.414	49 In Indium 113.488	50 Sn Tin 118.70	51 Sb Antimony (116)	52 Te Tellurium (126)	53 I Iodine (129)	54 Xe Xenon (131)		
55 Cs Cesium 132.91084	56 Ba Barium 137.327	57 - 71 Lanthanoids 138.90547	72 Hf Hafnium 174.949	73 Ta Tantalum 180.948	74 W Tungsten 183.944	75 Re Ruthenium 186.207	76 Os Osmium 190.23	77 Ir Iridium 192.277	78 Pt Platinum 190.964	79 Au Gold 196.96669	80 Hg Mercury 204.248	81 Tl Thallium 204.248	82 Pb Lead (207)	83 Bi Bismuth (209)	84 Po Polonium (209)	85 At Astatine (210)	86 Rn Radon (222)		
87 Fr Francium (228)	88 Ra Radium (226)	89 - 103 Actinoids (227)	104 Rf Rutherfordium (267)	105 Db Dubnium (268)	106 Sg Seaborgium (269)	107 Bh Bohrium (270)	108 Hs Hassium (269)	109 Mt Meitnerium (270)	110 Ds Darmstadtium (281)	111 Rg Roentgenium (282)	112 Cn Copernicium (285)	113 Nh Nihonium (286)	114 Fl Florium (289)	115 Mc Moscovium (290)	116 Lv Livermorium (291)	117 Ts Tennessee (291)	118 Og Oganesson (294)		

57 La Lanthanum 138.90547	58 Ce Cerium 140.90761	59 Pr Praseodymium 141.90761	60 Nd Neodymium 144.9242	61 Pm Promethium (145)	62 Sm Samarium 150.936	63 Eu Europium 151.944	64 Gd Gadolinium 157.9325	65 Tb Terbium 158.92535	66 Dy Dysprosium 162.500	67 Ho Holmium 164.93033	68 Er Erbium 167.228	69 Tm Thulium 168.93422	70 Yb Ytterbium 173.045	71 Lu Lutetium 174.9668
89 Ac Actinium (227)	91 Th Thorium 232.0377	91 Pa Protactinium 231.03688	92 U Uranium 238.02899	93 Np Neptunium (237)	94 Pu Plutonium (238)	95 Am Americium (243)	96 Gd Curium (247)	97 Bk Berkelium (247)	98 Cf Californium (250)	99 Es Einsteinium (252)	100 Fm Fermium (257)	101 Md Mendelevium (258)	102 No Nobelium (259)	103 Lr Lawrencium (266)

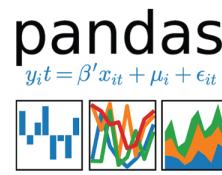


# Tools



## Data Extraction

Web scraping via  
Beautiful Soup  
Selenium



## Cleaning

Numpy  
Pandas



## Modeling

K Nearest-Neighbors  
Logistic Regression  
Random Forest  
Gaussian Support Vector Machines  
XGBoostClassifier



## Visualization

Tableau



# Methodology

Metric: fbeta\_score (beta= 0.5)

Scoring: Precision to minimize false positives.

Micro Averages: Aggregate the contributions of all classes to compute the average metric.

Validation: KFold(cv=5)

Step  
**01**

**Running Selected Models with Default Parameters**

Step  
**02**

**Balanced the Training Data via Adasyn/Smote**

Step  
**03**

**Tune the Hyperparameters of the Model**

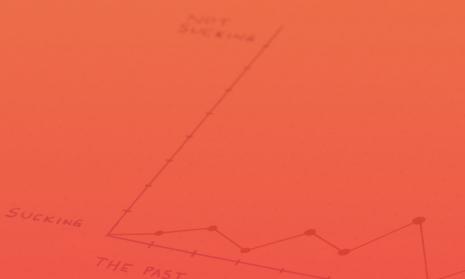
Step  
**04**

**Compared the Validation Scores Across Best Scores of Each Model**

Step  
**05**

**Chose Model Based on Score, Interpretability & Cost**

# Model Results



XGBoost Classifier →

Initial (Default)  
Initial + SMOTE  
Initial + Adasyn  
GridSearch + SMOTE  
GridSearch + Adasyn

## Best Validation Scores From Each Model

Model	K Nearest Neighbors	Logistic Regression	Random Forest	SVM	XGBoost Classifier
Method	SMOTE + Gridsearch	SMOTE + Gridsearch	ADASYN + Gridsearch	SMOTE + Default Param	ADASYN + Gridsearch
Validation Score	0.964	0.945	0.975	0.944	0.976
Predict Score	0.825	0.835	0.864	0.870	0.874

# Logistic Regression & Coefficients



Top 5

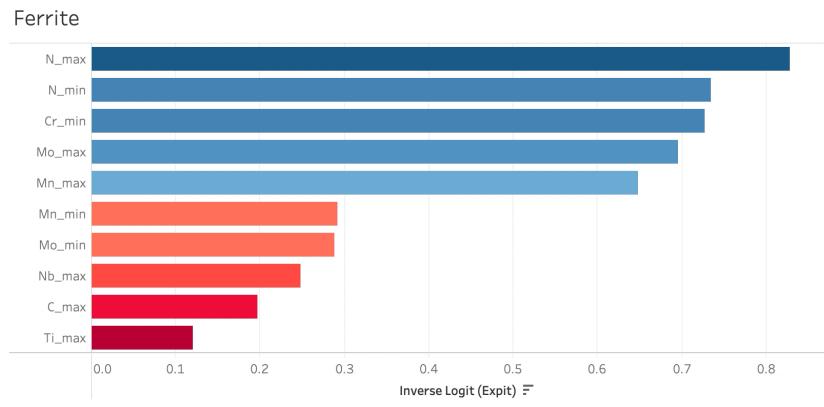
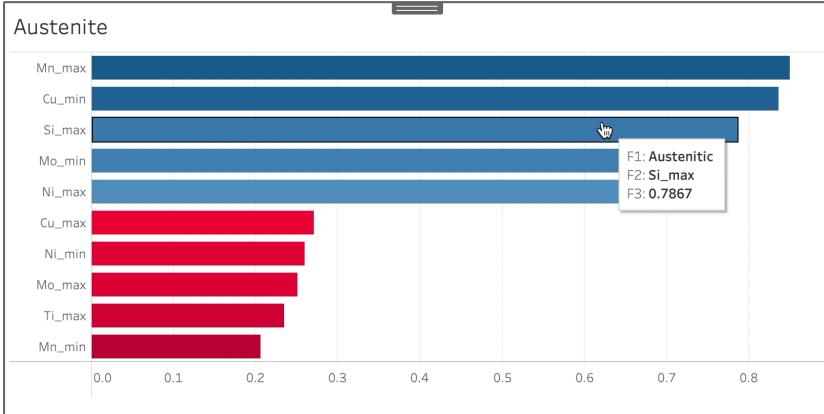
Bottom 5

Top 5

Bottom 5

SUCKING

THE PAST



# Conclusion

All models did well, but XGBoost performed the best

Balanced models perform better in classification

## Future Work

Add more steel grades and add mechanical properties of steel

Perform a similar model with chemicals & polymers if dataset is available

# Contact



jamesleefan@gmail.com



<https://github.com/jimmfan/>



www.website.com

# References

## Python

```
def kfold_test(model, X_train, y_train, n_val=5, shuffle_val=True, random_val=42,
average_val='micro', beta_val = 0.5):
    kf = KFold(n_splits=n_val, shuffle=shuffle_val, random_state=random_val)
    model_results = []
    for train_ind, val_ind in kf.split(X_train,y_train):

        X_train_k, y_train_k = X_train[train_ind], y_train[train_ind]
        X_val, y_val = X_train[val_ind], y_train[val_ind]

        model.fit(X_train_k, y_train_k)
        model_results.append(fbeta_score(model.predict(X_val), y_val, average=average_val,
beta=beta_val))
    return model_results
```

## References (cont)

```
# Logistic Regression  
  
#GridSearch  
from scipy.stats import uniform  
# Create regularization penalty space  
penalty = ['l1', 'l2']  
  
# Create regularization hyperparameter distribution using uniform distribution  
C = uniform(loc=0, scale=50)  
  
# Create hyperparameter options  
hyperparameters = dict(C=C, penalty=penalty)
```

# References (cont)

```
# SMOTED training set
grid_srch = RandomizedSearchCV(model, hyperparameters, n_iter=50, cv=5, scoring='precision_micro', iid=False)
grid_srch.fit(X_smoted, y_smoted)
print(grid_srch.best_score_)
param = grid_srch.best_params_
# {'C': 39.756662587546835, 'penalty': 'l2'}

model = LogisticRegression(
    penalty=param['penalty'],
    dual=False,
    tol=0.0001,
    C=param['C'],
    max_iter=100,
    multi_class='ovr',
    n_jobs=-1)
gridsearch_res = kfold_test(model, X_smoted, y_smoted, n_val=5, shuffle_val=True, random_val=42)
np.mean(gridsearch_res)

0.9450777202072539

model.fit(X_smoted, y_smoted)
fbeta_score(model.predict(X_test), y_test, average='micro', beta=0.5)
```