MBTI PREDICTOR 5000

Cady Li
Dhruval Kothari
Jonathan Tay

TABLE OF CONTENTS

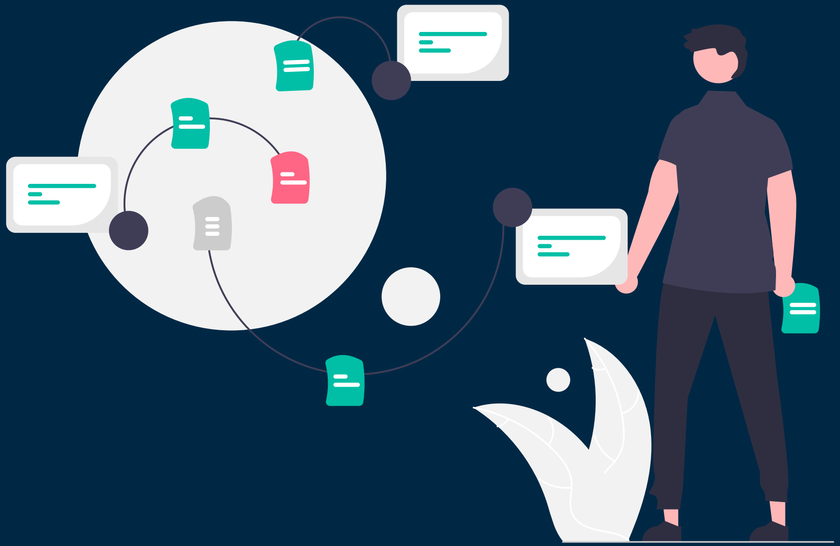
OVERVIEW

WHAT DOES IT DO?

01

01 OVERVIEW

AN INFORMATION GROWTH



Information growth has accelerated with the advent of social media especially in the form of textual data types.

01

OVERVIEW

POSSIBLE APPLICATIONS

Team formation

Job Applications

Relationship Apps

Online Marketing

01 OVERVIEW

MYERS-BRIGGS TYPE INDICATOR

E — Extroverts
are energized by people, enjoy a variety of tasks, a quick pace, and are good at multitasking.

I — Introverts
often like working alone or in small groups, prefer a more deliberate pace, and like to focus on one task at a time.

S — Sensors
are realistic people who like to focus on the facts and details, and apply common sense and past experience to come up with practical solutions to problems.

N — Intuitives
prefer to focus on possibilities and the big picture, easily see patterns, value innovation, and seek creative solutions to problems.

T — Thinkers
tend to make decisions using logical analysis, objectively weigh pros and cons, and value honesty, consistency, and fairness.

F — Feelers
tend to be sensitive and cooperative, and decide based on their own personal values and how others will be affected by their actions.

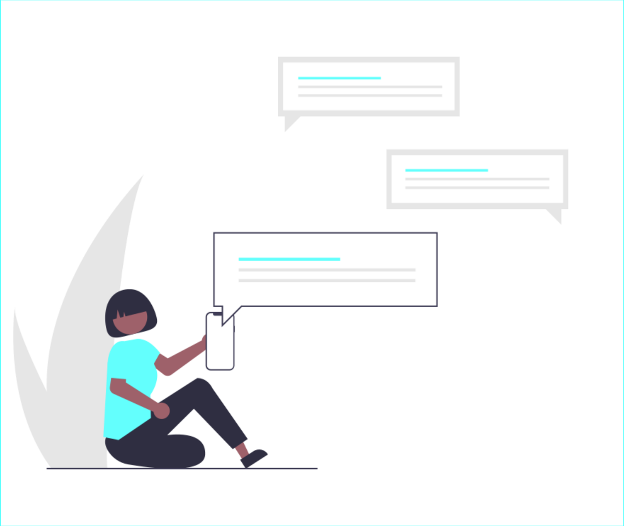
J — Judgers
tend to be organized and prepared, like to make and stick to plans, and are comfortable following most rules.

P — Perceivers
prefer to keep their options open, like to be able to act spontaneously, and like to be flexible with making plans.

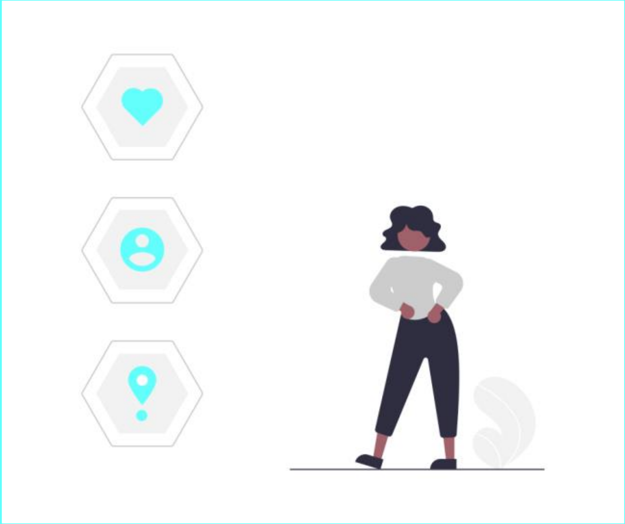
01 OVERVIEW

OUR MODEL

Textual Data



MBTI

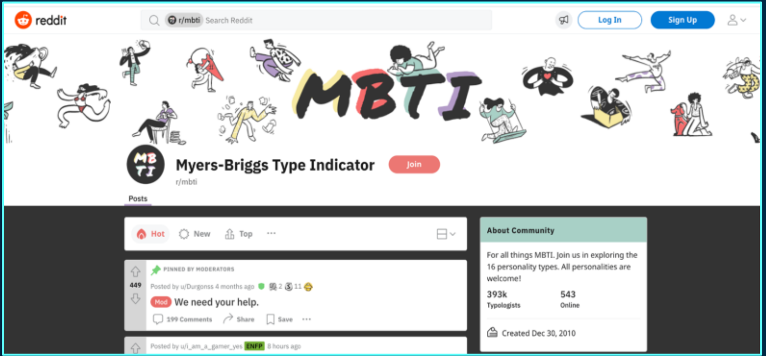


Predict personality type from textual data

01 DATA RETRIEVAL

Reddit: All comments from users in r/mbti

```
SELECT  flair_text.author_flair_text as flair_text, comments.body as body,
comments.subreddit as subreddit, comments.author as author FROM
  (
  SELECT author,author_flair_text
  FROM [fh-bigquery:reddit_comments.all]
  WHERE author_flair_text != 'null'
  AND REGEXP_MATCH(author_flair_text,r'([IEie][SNsn][TFtf][JPjp]\W)')
  GROUP BY author,author_flair_text
  ) AS flair_text
INNER JOIN
  (
  SELECT  author_flair_text, body, subreddit, author
  FROM [fh-bigquery:reddit_comments.all]
  ) AS comments
ON
comments.author = flair_text.author
```



July 30, 2018

Dataset Open Access

Myers Briggs Personality Tags on Reddit Data

Dylan Storey

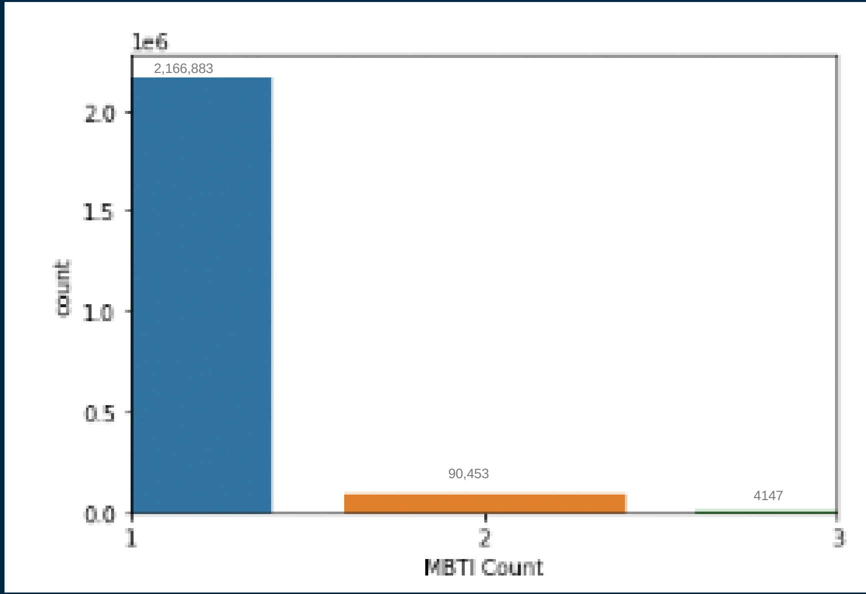
This data was pulled on 11/10/2018 from google big query using the following query:

DATA VISUALIZATION, CLEANING & PREPROCESSING

02

02 DATA CLEANING

REMOVING AMBIGUOUS DATA




```
df['MBTIs'] = df.apply(lambda row: re.findall("[IiEe][SsNn][FfTt]
              [PpJj]",row.flair_text), axis=1)
df['MBTI Count'] = df['MBTIs'].str.len()
sns.countplot(x = 'MBTI Count', data = df).set_xlim(0,2)
```

Plotting box plot for MBTI count per user

```
author_to_remove = df[df['MBTIs'].str.len() > 1]['author'].tolist()
df1 = df[~df['author'].isin(author_to_remove)].copy()
```

Removing users with more than one MBTI

02

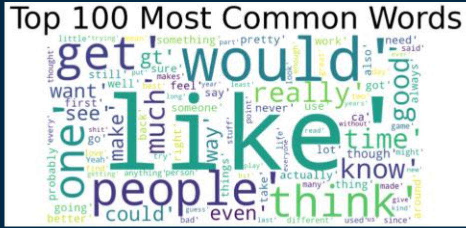
DATA CLEANING

REMOVING STOPWORDS

```python
def preprocess_text(sentence):
    sentence = str(sentence)
    sentence = p.clean(sentence)
    #tokenize word and remove stop words
    word_tokens = word_tokenize(sentence)
    sentence = [w for w in word_tokens if not w.lower() in stop_words]
    sentence = [x for x in sentence if "'" not in x]
    sentence = ' '.join(sentence)
    # Removing multiple spaces
    sentence = re.sub(r"\s+", " ", sentence)
    sentence = sentence.strip()
    sentence = sentence.replace('\n',' ')
    return sentence

 df1['clean'] = df1['body'].apply(lambda x:preprocess_text(x))
```

Removing stop words gives more **context** to the comments and removes **"noisy"** words that do not add value to our prediction

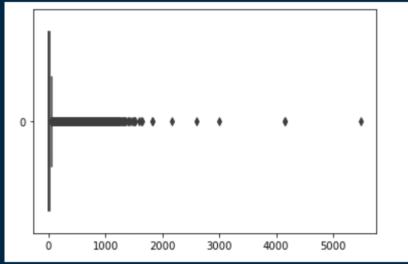
Before removing stopwords

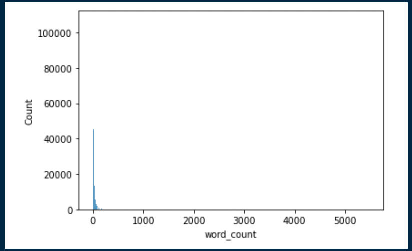


After removing stopwords

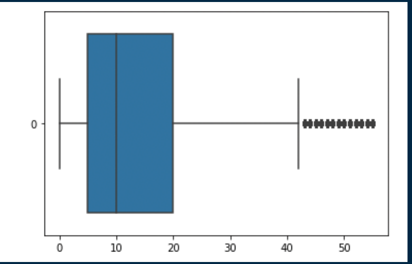
02 DATA CLEANING

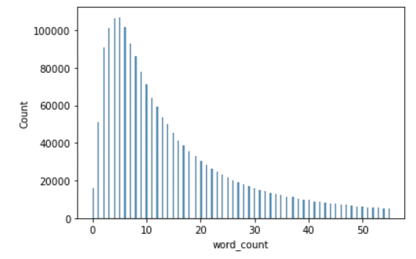
NORMALIZING COMMENT LENGTHS





Before removal of outliers





After removal of outliers

```
df1['word_count'].describe()
```

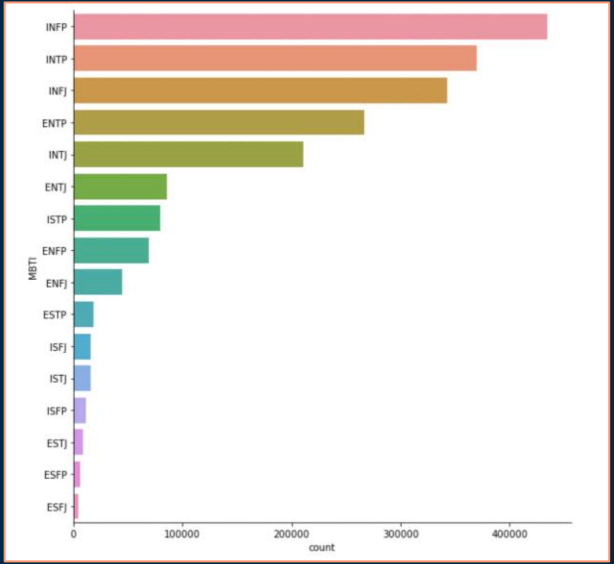
```
count    1984338.00000
mean          24.67303
std           44.81360
min            0.00000
25%            6.00000
50%           12.00000
75%           26.00000
max         5485.00000
Name: word_count, dtype: float64
```

We limit comment lengths to between 20-30 words due to limited processing power and considering mean and median comment lengths

02 DATA CLEANING

DOWN SAMPLING CLASSES

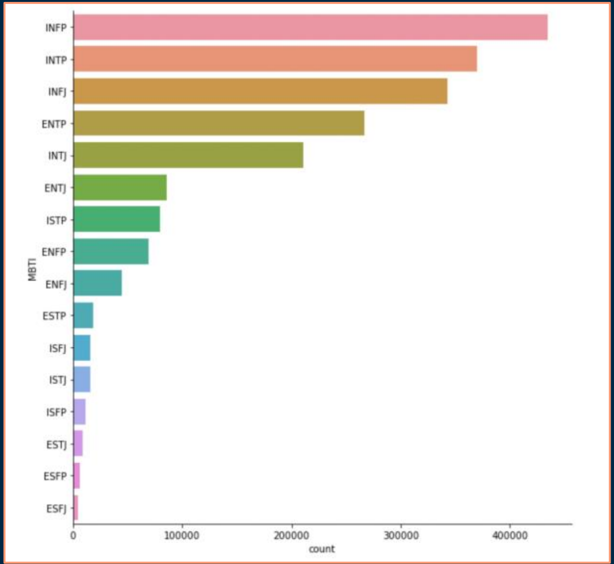
There are unequal proportions of each MBTI type



Before

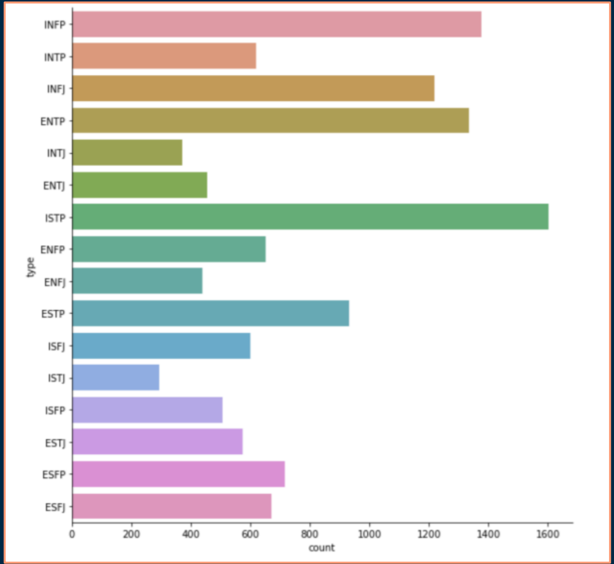
02 DATA CLEANING

DOWN SAMPLING CLASSES



Before

1,984,338 samples



After

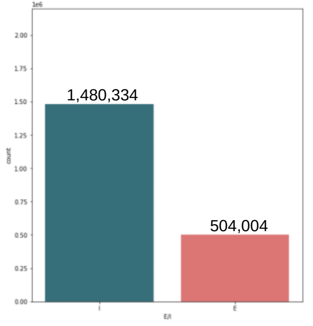
12,356 samples

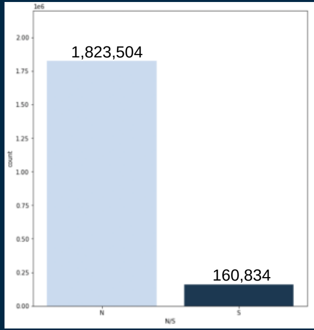
02 DATA CLEANING

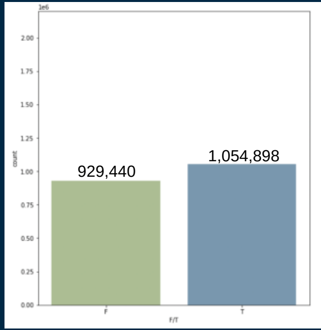
DOWN SAMPLING CLASSES

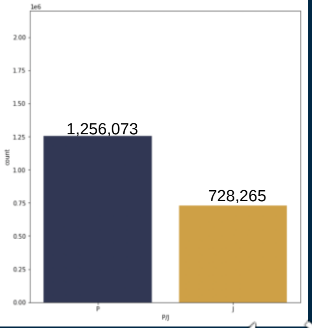
There is a wide distribution of each class and counts are largely unequal which may lead to the model over-fitting to the majority class

Before





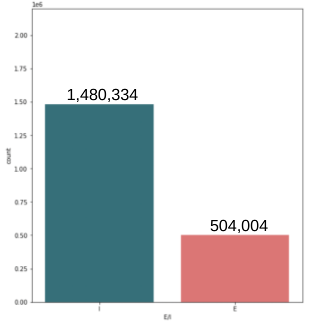


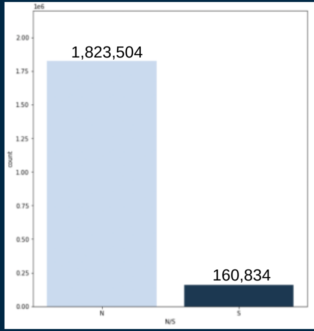


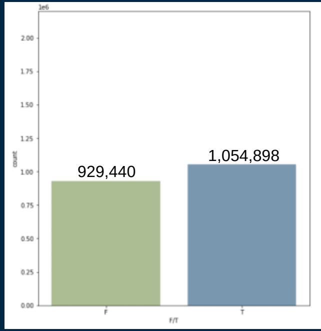
02 DATA CLEANING

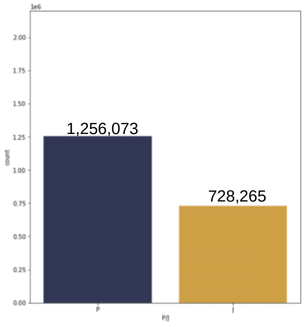
DOWN SAMPLING CLASSES

Before

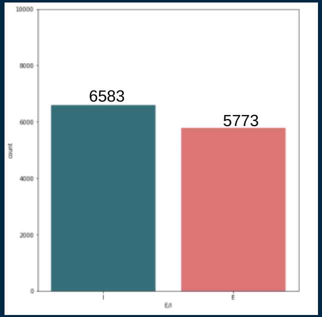


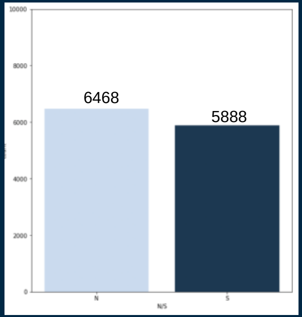


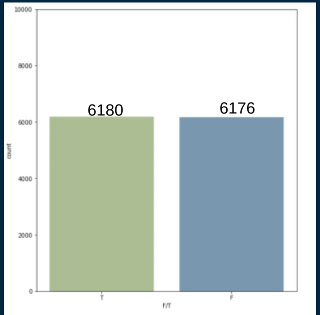


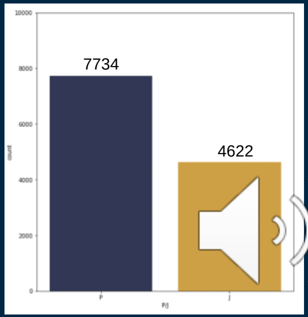


After









02

DATA CLEANING

DOWN SAMPLING CLASSES

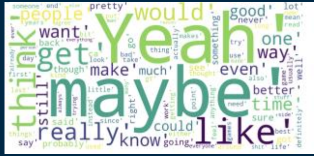
```
distrib = dict(allCounts)

counts = {}
for i in distrib:
    for j in i:
        c = counts.get(j,0)
        counts[j] = c+distrib[i]
ie = counts["I"] - counts["E"]

total = distrib["INTP"] + distrib["INFP"] + distrib["INTJ"] +
distrib["INFJ"] +distrib["ISTP"] + distrib["ISFP"] + distrib["ISTJ"] +
distrib["ISFJ"]
for i in distrib:
    if i[0] == "I":
        toremove = int(ie*(distrib[i]/total))
        distrib[i] -= toremove
```

02

DATA VISUALIZATION

WORDCLOUD

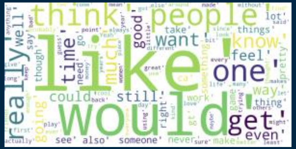
ISFJ



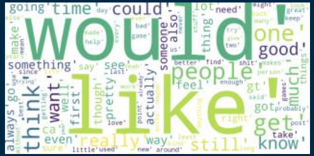
ISFP

ISTJ

ISTP



INFJ

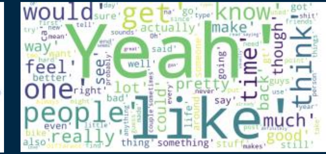


INFP

INTP

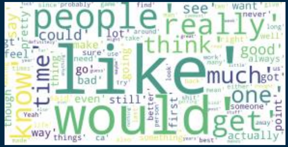
INTP

ESFJ



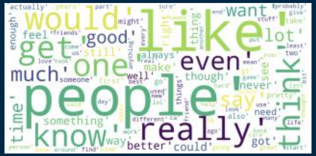
ESFP

ESTJ



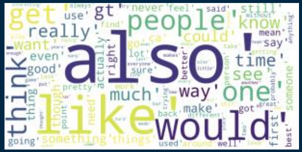
ESTP

ENFJ



ENFP

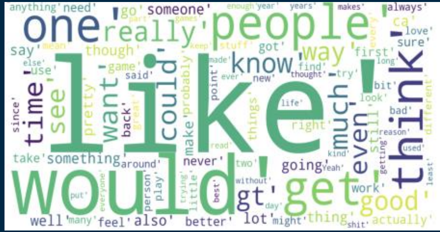
ENTJ



ENTP

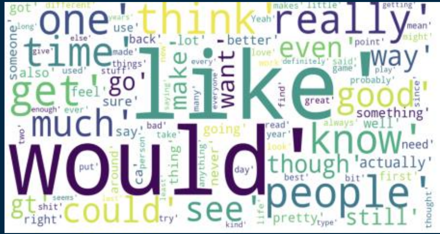
02 DATA VISUALIZATION

WORDCLOUD



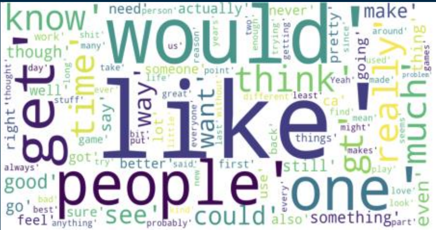
Introvert

Extrovert

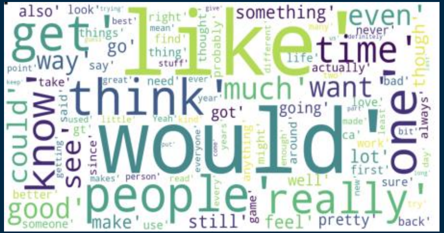


Sensors

Intuitives



Thinking



Feeling

Perceiver

Judger

03

MACHINE LEARNING

WHAT DID WE USE?

03 MACHINE LEARNING

TOKENIZER

BERT: Bidirectional Encoder Representation From Transformers

- Pretrained on unsupervised Wikipedia and BookCorpus Datasets using language modelling.

ALBERT: A lite BERT that lowers memory consumption and increases the training speed of BERT

- Learns contextual relations between words in a text

Transformer based approach is superior to the standard LSTM approaches and deeply bidirectional

03 MACHINE LEARNING

TOKENIZER --> Converts words to vectors

	text
0	post ` try telling people time always joking ...
1	kind strange lump vague references religions o...
2	helped friends mine purchase two Honda Civic S...
3	, one come back ? , indeed , need meds , chang...
4	promised fuck climate campaign . promises fuck...
...	...
131335	EKIN . forget exact number . class small fills...
131336	Congrats ! ! senior , makes nostalgic sad , ex...
131337	Oh called . sanguine choleric * , whatever mea...
131338	traditional calendar ; Extraordinary Form pari...
131339	feel pain , friend . Craigslist crap shoot , e...

131340 rows × 1 columns

→

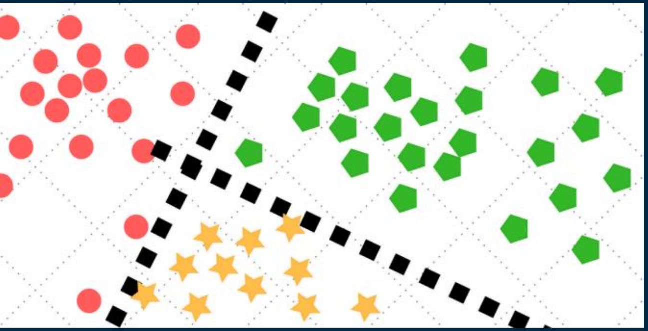
	0	1	2	3	4	5	6
0	0.236103	-0.250498	0.364666	1.068560	0.058454	-0.560975	1.405643
1	-0.910153	0.755426	0.616445	0.529585	-0.573597	-0.533925	1.681382
2	-1.336975	-0.329202	1.147945	-1.601125	-0.038263	-0.191345	2.353152
3	0.852663	-1.131741	0.702313	0.933792	0.787668	0.265421	-1.255930
4	0.641273	0.661787	-0.949508	0.768758	-0.842885	0.330161	1.045339
...	...	...	...	...	...	...	...
131335	0.579979	1.240591	0.911377	-0.035959	0.736116	1.456866	0.607049
131336	0.070313	-0.476752	-0.300813	2.849004	1.418178	-1.435400	-0.612279
131337	0.087869	0.627987	-0.840733	0.125208	-0.803629	-0.430973	0.480788
131338	0.171308	-1.423708	0.447174	-0.527545	0.777886	0.094312	0.152559
131339	0.261051	0.582918	-0.841849	0.666375	-0.673524	0.883894	-1.360341

131340 rows × 768 columns

03

MACHINE LEARNING

CLASSIFIERS - sklearn



1. K Nearest Neighbors
2. Linear SVM (Support Vector Machine)
3. Decision Tree Classifier
4. Neural Net (Multi Layer Perceptron)
5. Random Forest
6. RBF (Radial Basis Function) SVM
7. Naïve Bayes (Quadratic Discriminant Analysis)
8. AdaBoost

03 MACHINE LEARNING

Process

```
data_x, data_y = get_inputs(filename, layer)
data_x = StandardScaler().fit_transform(data_x)
x_train, x_test, real_y_train, real_y_test = train_test_split(data_x, data_y, test_size = 0.2)

labels = ["Extraversion (E) vs Introversion (I)", "Intuition (N) vs Sensing (S)", "Feeling (F) vs
Thinking (T)", "Judging (J) vs Perceiving (P)" ]

for idx, label in enumerate(labels):
    y_train = real_y_train[:, idx]
    y_test = real_y_test[:, idx]
    for clf in classifiers:
        clf.fit(x_train, y_train)
        score = clf.score(x_test, y_test)
        y_pred = clf.predict(x_test)

        conf_matrix = confusion_matrix(y_test, y_pred)
        tn, fp = conf_matrix[0]
        fn, tp = conf_matrix[1]

        precision = tp / (tp + fp)
        recall = tp / (tp+fn)
        accuracy = (tp + tn) / (tp+tn+fp+fn)
        f1 = (2 * (precision * recall)) / (precision + recall)
```

- Train Test Split of 0.2
- Select axis to train on for binary classification
- Fit model from train data
- Plot Confusion Matrix and calculate metrics

03 MACHINE LEARNING

WHAT WE USED

Techniques from Course	New Techniques tried
Sampling and Preprocessing	One-Hot Encoding, Down Sampling
Decision Tree Classifier	Word Vectorization and other NLP techniques
Confusion Matrices	Other classifiers (e.g. SVM)

RESULTS & ACCURACY

04

HOW WELL DID IT DO?

04 RESULTS & ACCURACY

Extraversion (E) vs Introversion (I)

Analysis of Extraversion (E) vs Introversion (I)

Model	Accuracy	F1 Score	Precision	Recall
K Nearest Neighbors	0.517	0.484	0.482	0.487
Linear SVM	0.537	0.467	0.504	0.436
Decision Tree	0.515	0.443	0.477	0.414
Neural Net (MLP)	0.530	**0.500**	0.496	**0.504**
Random Forest	0.542	0.483	0.509	0.460
RBF SVM	**0.558**	0.100	**0.968**	0.053
Naive Bayes	0.538	0.495	0.505	0.487
AdaBoost	0.529	0.453	0.493	0.419

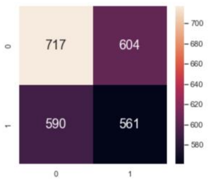
The best models for E/I are RBF SVM and Neural Net.

04

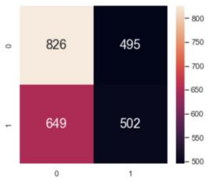
RESULTS & ACCURACY

Extraversion (E) vs Introversion (I)

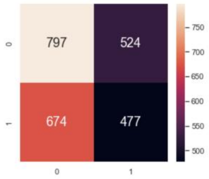
```
Trying out: Nearest Neighbors
Labelling: Extraversion (E) vs Introversion (I)

Classification Accuracy:          0.5169902912621359
F1 score: 0.4844559585492228
Precision: 0.4815450643776824
Recall: 0.48740225890529976
Accuracy: 0.5169902912621359
Took 89 seconds
```




```
Trying out: Linear SVM
Labelling: Extraversion (E) vs Introversion (I)

Classification Accuracy:          0.5372168284789643
F1 score: 0.4674115456238361
Precision: 0.5035105315947843
Recall: 0.436142484795829 73
Accuracy: 0.5372168284789643
Took 254 seconds
```

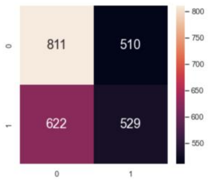



```
Trying out: Decision Tree
Labelling: Extraversion (E) vs Introversion (I)

Classification Accuracy:          0.5153721682847896
F1 score: 0.44330855018587356
Precision: 0.47652347652347654
Recall: 0.4144222415291051
Accuracy: 0.5153721682847896
Took 9 seconds
```

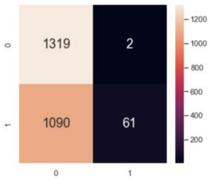



```
Trying out: Neural Net
Labelling: Extraversion (E) vs Introversion (I)

Classification Accuracy:          0.5303398058252428
F1 score: 0.4997845756139595
Precision: 0.49572649572649574
Recall: 0.5039096437880104
Accuracy: 0.5303398058252428
Took 10 seconds
```

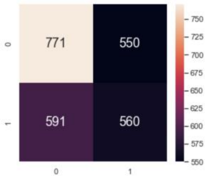

```
Trying out: Random Forest
Labelling: Extraversion (E) vs Introversion (I)

Classification Accuracy:          0.5420711974110033
F1 score: 0.4831050228310502
Precision: 0.5091434071222329
Recall: 0.4596003475238923
Accuracy: 0.5420711974110033
Took 0 seconds
```



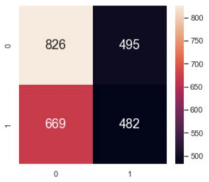

```
Trying out: RBF SVM
Labelling: Extraversion (E) vs Introversion (I)

Classification Accuracy:          0.558252427184466
F1 score: 0.10049423393739704
Precision: 0.9682539682539683
Recall: 0.052997393570807995
Accuracy: 0.558252427184466
Took 272 seconds
```




```
Trying out: Naive Bayes
Labelling: Extraversion (E) vs Introversion (I)

Classification Accuracy:          0.5384304207119741
F1 score: 0.495356037151702 84
Precision: 0.5045045045045045
Recall: 0.48653344917463076
Accuracy: 0.5384304207119741
Took 0 seconds
```




```
Trying out: AdaBoost
Labelling: Extraversion (E) vs Introversion (I)

Classification Accuracy:          0.529126213592233
F1 score: 0.4530075187969924
Precision: 0.49334698055271237
Recall: 0.41876629018245004
Accuracy: 0.529126213592233
Took 41 seconds
```



04

RESULTS & ACCURACY

Intuition (N) vs Sensing (S)

Analysis of Intuition (N) vs Sensing (S)

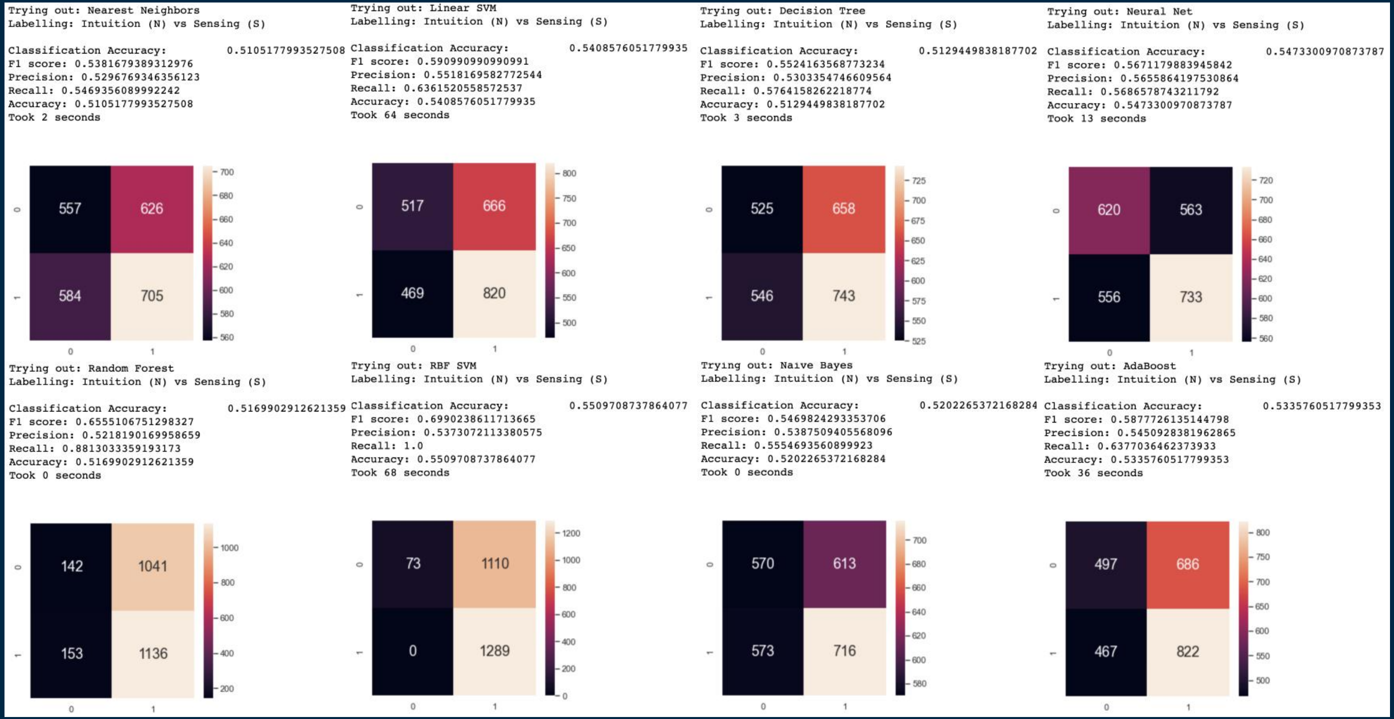
Model	Accuracy	F1 Score	Precision	Recall
K Nearest Neighbors	0.511	0.538	0.530	0.547
Linear SVM	0.541	0.591	**0.552**	0.636
Decision Tree	0.513	0.552	0.530	0.576
Neural Net (MLP)	0.547	0.567	0.566	0.569
Random Forest	0.517	0.656	0.522	0.881
RBF SVM	**0.551**	**0.699**	0.537	**1.0**
Naive Bayes	0.520	0.547	0.539	0.555
AdaBoost	0.534	0.588	0.545	0.638

The best models for N/S are RBF SVM and Neural Net.

04

RESULTS & ACCURACY

Intuition (N) vs Sensing (S)



04 RESULTS & ACCURACY

Feeling (F) vs Thinking (T)

Analysis of Feeling (F) vs Thinking (T)

Model	Accuracy	F1 Score	Precision	Recall
K Nearest Neighbors	0.524	0.521	0.511	0.531
Linear SVM	0.552	0.558	0.537	0.580
Decision Tree	0.524	0.537	0.510	0.568
Neural Net (MLP)	0.546	0.545	0.532	0.558
Random Forest	0.540	0.544	0.525	0.564
RBF SVM	0.505	**0.662**	0.496	**0.995**
Naive Bayes	**0.555**	0.517	**0.548**	0.489
AdaBoost	0.530	0.536	0.516	0.557

The best models for F/T are RBF SVM and Naïve Bayes.

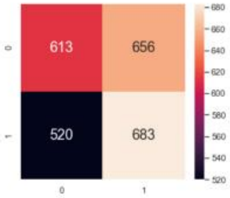
04 RESULTS & ACCURACY

Feeling (F) vs Thinking (T)

```
Trying out: Nearest Neighbors
Labelling: Feeling (F) vs Thinking (T)

Classification Accuracy:          0.5242718446601942
F1 score: 0.5207823960880196
Precision: 0.5107913669064749
Recall: 0.5311720698254364
Accuracy: 0.5242718446601942
Took 114 seconds
```

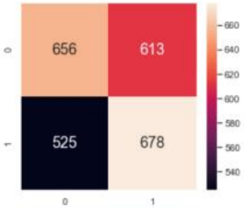

```
Trying out: Linear SVM
Labelling: Feeling (F) vs Thinking (T)

Classification Accuracy:          0.5521844660194175
F1 score: 0.5577307231322414
Precision: 0.536923076923077
Recall: 0.5802161263507897
Accuracy: 0.5521844660194175
Took 261 seconds
```


```
Trying out: Decision Tree
Labelling: Feeling (F) vs Thinking (T)

Classification Accuracy:          0.5242718446601942
F1 score: 0.5373721479150274
Precision: 0.5100821508588499
Recall: 0.5677472984206151
Accuracy: 0.5242718446601942
Took 4 seconds
```

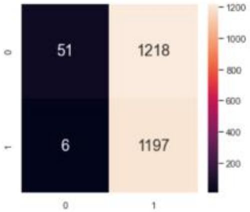



```
Trying out: Neural Net
Labelling: Feeling (F) vs Thinking (T)

Classification Accuracy:          0.5461165048543689
F1 score: 0.5446428571428572
Precision: 0.5321173671689136
Recall: 0.5577722360764755
Accuracy: 0.5461165048543689
Took 26 seconds
```

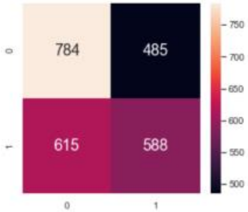

```
Trying out: Random Forest
Labelling: Feeling (F) vs Thinking (T)

Classification Accuracy:          0.5396440129449838
F1 score: 0.5437048917401764
Precision: 0.5251742835011619
Recall: 0.5635910224438903
Accuracy: 0.5396440129449838
Took 0 seconds
```



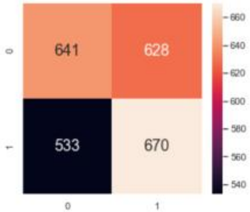

```
Trying out: RBF SVM
Labelling: Feeling (F) vs Thinking (T)

Classification Accuracy:          0.5048543689320388
F1 score: 0.6616915422885573
Precision: 0.4956521739130435
Recall: 0.9950124688279302
Accuracy: 0.5048543689320388
Took 255 seconds
```




```
Trying out: Naive Bayes
Labelling: Feeling (F) vs Thinking (T)

Classification Accuracy:          0.555016181
F1 score: 0.5166959578207381
Precision: 0.5479962721342032
Recall: 0.48877805486284287
Accuracy: 0.5550161812297735
Took 0 seconds
```




```
Trying out: AdaBoost
Labelling: Feeling (F) vs Thinking (T)

Classification Accuracy:          0.5303398058252428
F1 score: 0.5357856857257097
Precision: 0.5161787365177196
Recall: 0.5569409808811305
Accuracy: 0.5303398058252428
Took 46 seconds
```



04

RESULTS & ACCURACY

Perceiving (P) vs Judging (J)

Analysis of Perceiving (P) vs Judging (J)

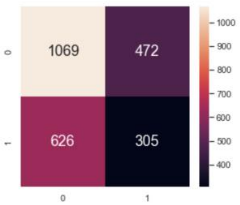
Model	Accuracy	F1 Score	Precision	Recall
K Nearest Neighbors	0.556	0.357	0.392	0.327
Linear SVM	0.621	0.150	0.482	0.089
Decision Tree	0.617	0.063	0.395	0.034
Neural Net (MLP)	0.565	0.408	0.418	0.398
Random Forest	0.623	0.002	0.500	0.001
RBF SVM	**0.631**	0.044	**0.955**	0.023
Naive Bayes	0.493	**0.444**	0.378	**0.537**
AdaBoost	0.596	0.204	0.395	0.137

The best models for P/J are RBF SVM and Naïve Bayes.

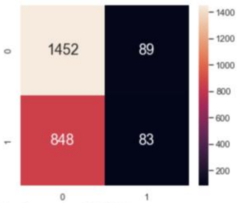
04 RESULTS & ACCURACY

Perceiving (P) vs Judging (J)

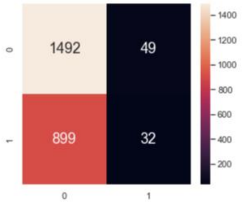
```
Trying out: Nearest Neighbors
Labelling: Judging (J) vs Perceiving (P)

Classification Accuracy:          0.5558252427184466
F1 score: 0.35714285714285 72
Precision: 0.392535392535392 55
Recall: 0.3276047261009667
Accuracy: 0.5558252427184466
Took 95 seconds
```



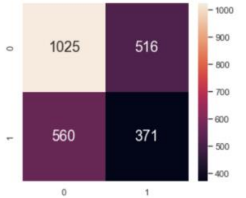

```
Trying out: Linear SVM
Labelling: Judging (J) vs Perceiving (P)

Classification Accuracy:          0.6209546925566343
F1 score: 0.15049864007252944
Precision: 0.48255813953488375
Recall: 0.08915145005370569
Accuracy: 0.6209546925566343
Took 225 seconds
```



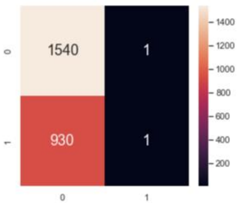

```
Trying out: Decision Tree
Labelling: Judging (J) vs Perceiving (P)

Classification Accuracy:          0.616504854368932
F1 score: 0.06324110671936758
Precision: 0.3950617283950617
Recall: 0.034371643394199784
Accuracy: 0.616504854368932
Took 3 seconds
```



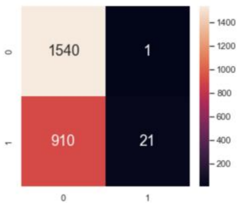

```
Trying out: Neural Net
Labelling: Judging (J) vs Perceiving (P)

Classification Accuracy:          0.5647249190938511
F1 score: 0.4081408140814082
Precision: 0.4182638105975197
Recall: 0.39849624060150374
Accuracy: 0.5647249190938511
Took 8 seconds
```



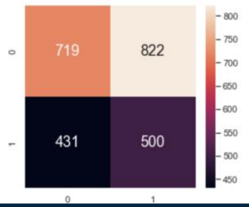

```
Trying out: Random Forest
Labelling: Judging (J) vs Perceiving (P)

Classification Accuracy:          0.6233818770226537
F1 score: 0.0021436227224008574
Precision: 0.5
Recall: 0.0010741138560687433
Accuracy: 0.6233818770226537
Took 0 seconds
```



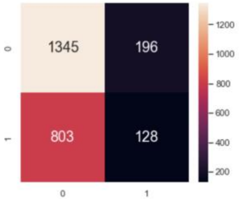

```
Trying out: RBF SVM
Labelling: Judging (J) vs Perceiving (P)

Classification Accuracy:          0.6314724919093851
F1 score: 0.04407135362014691
Precision: 0.9545454545454546
Recall: 0.022556390977443608
Accuracy: 0.6314724919093851
Took 273 seconds
```




```
Trying out: Naive Bayes
Labelling: Judging (J) vs Perceiving (P)

Classification Accuracy:          0.4931229773462783
F1 score: 0.44385264092321347
Precision: 0.37821482602118
Recall: 0.5370569280343717
Accuracy: 0.4931229773462783
Took 0 seconds
```




```
Trying out: AdaBoost
Labelling: Judging (J) vs Perceiving (P)

Classification Accuracy:          0.595873786407767
F1 score: 0.20398406374501 99
Precision: 0.3950617283950617
Recall: 0.13748657357679914
Accuracy: 0.595873786407767
Took 39 seconds
```

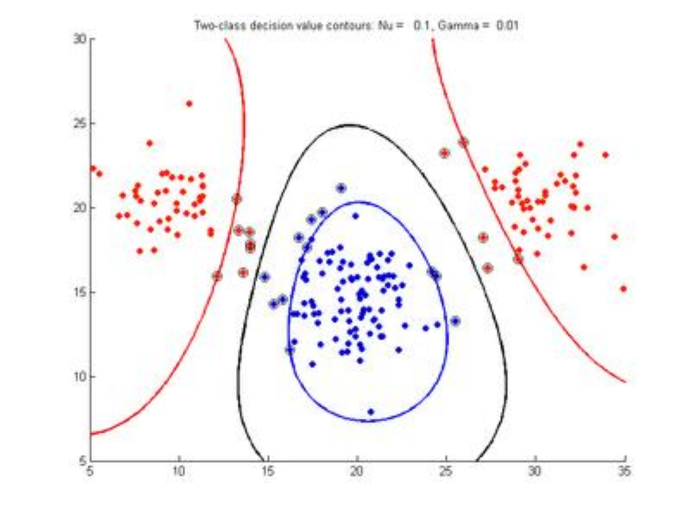


INSIGHTS & FUTURE RECOMMENDATIONS

05

DATA-DRIVEN INSIGHTS &
WHATS NEXT?

05 INSIGHTS



No issue with classifiers as we tried a range of classifiers but all of them did not improve the accuracy above a bound of 65%.

We recommend the RBF SVM as it is the best performing classifier, with the highest average accuracy

05

FUTURE RECOMMENDATIONS

TRAIN OUR MODEL FROM OTHER TEXT SOURCES

Tweets

Personal Profile Captions

Personal Statements

Use primary text data instead of secondary text data.

05

FUTURE RECOMMENDATIONS

DIFFERENT MODELS

OpenAI GPT-3

Trained on billions of parameters 470 times bigger than BERT model

XLNet

Uses auto-regressive (AR) models instead of auto-encoding (AE) which has improved accuracy on tasks like natural language inference

05 FUTURE RECOMMENDATIONS:
HYPERPARAMETER TUNING

Hyperparameters help us find the balance between overfitting and underfitting our model.

E.g. RBF SVM:
We can optimize the C and Gamma parameters.

THANK YOU