Using Machine Learning to Predict Citation Metrics

Bercin Cenik & Tina Huang

Qualifications for a job in academic science

- Funding status
- Commitment to research
- Commitment to teaching
- Educational background

What faculty hiring committees want

Charles B Wright & Nathan L Vanderford

PhD trainees aspiring to become faculty need to know the credentials search committees value most in an applicant.

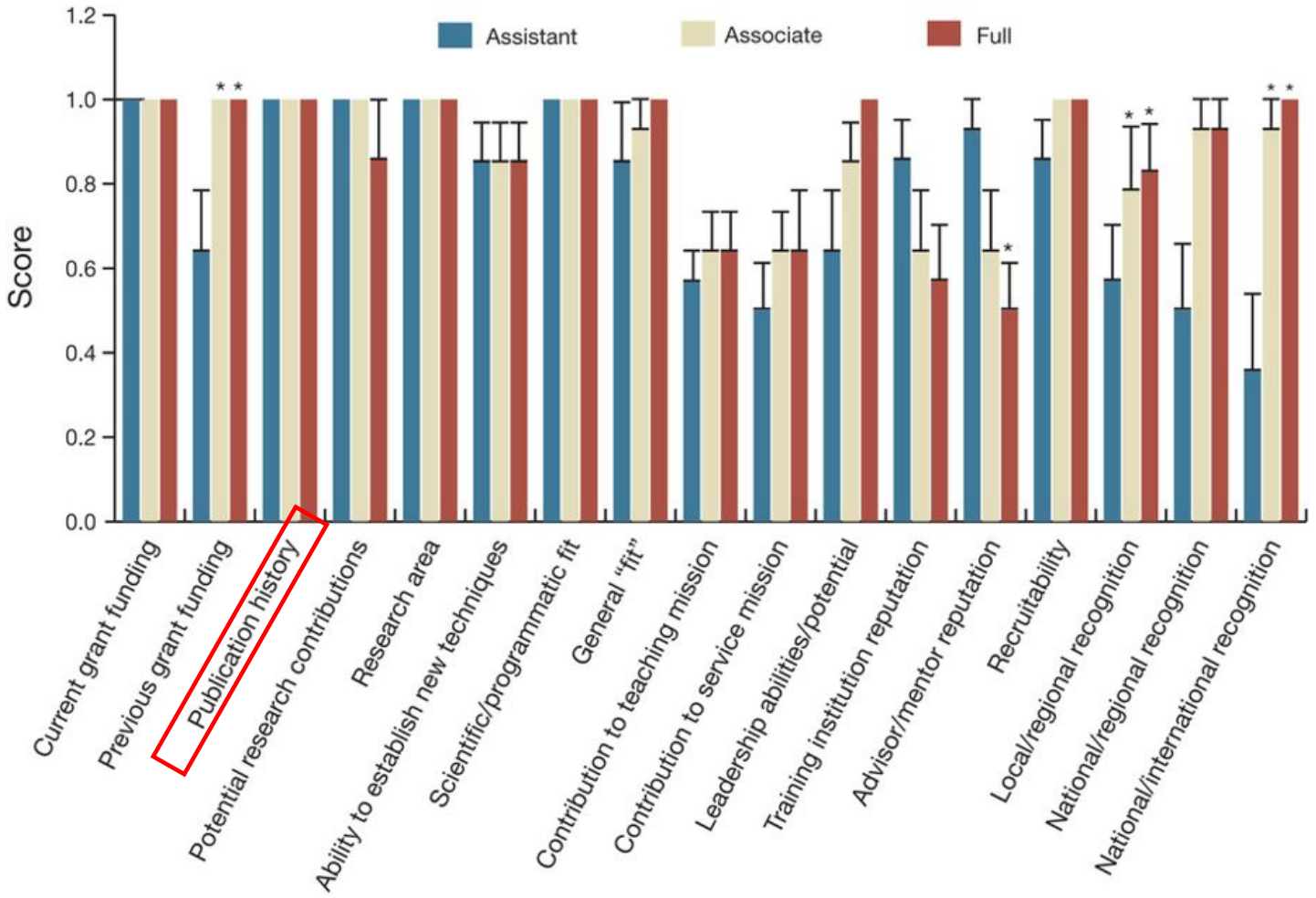


Table 1 Faculty credentials fall into four categories

	Assistant	Associate	Full
Core competencies			
Current grant funding	1.00	1.00	1.00
Publication history	1.00	1.00	1.00
Research area	1.00	1.00	1.00
Scientific/programmatic fit	1.00	1.00	1.00
Potential research contributions	1.00	1.00	0.86
Ability to establish new techniques	0.86	0.86	0.86
General fit	0.86	0.93	1.00
Recruitability	0.86	1.00	1.00
Initial necessities			
Advisor/mentor reputation	0.93	0.64	0.50
Training institution reputation	0.86	0.64	0.57
Necessities for advancement			
National/international recognition	0.36	0.93	1.00
Regional/national recognition	0.50	0.93	0.93
Previous grant funding	0.64	1.00	1.00
Leadership abilities/potential	0.64	0.86	1.00
Unnecessary credentials			
Contribution to service mission	0.50	0.64	0.64
Contribution to teaching mission	0.57	0.64	0.64
Local/regional recognition	0.57	0.79	0.83

Credential scores followed one of four distinct trends: core competencies, initial necessities, necessities for advancement, and unnecessary credentials.

Scientific citation is providing detailed reference in a scientific publication, typically a paper or book, to previous published (or occasionally private) communications which have a bearing on the subject of the new publication. The purpose of citations in original work is to allow readers of the paper to refer to cited work to assist them in judging the new work, source background information vital for future development, and acknowledge the contributions of earlier workers. Citations in, say, a review paper bring together many sources, often recent, in one place.

To a considerable extent the quality of work, in the absence of other criteria, is judged on the number of citations received, adjusting for the volume of work in the relevant topic.[*citation needed*] While this is not necessarily a reliable measure, counting citations is trivially easy; judging the merit of complex work can be very difficult.[*citation needed*]

Reference section in scientific paper.

Citation impact quantifies the citation usage of scholarly works
Citation counts for an author, an individual article, or a publication

Websites that generate citation metrics

Scopus®

WEB OF SCIENCE

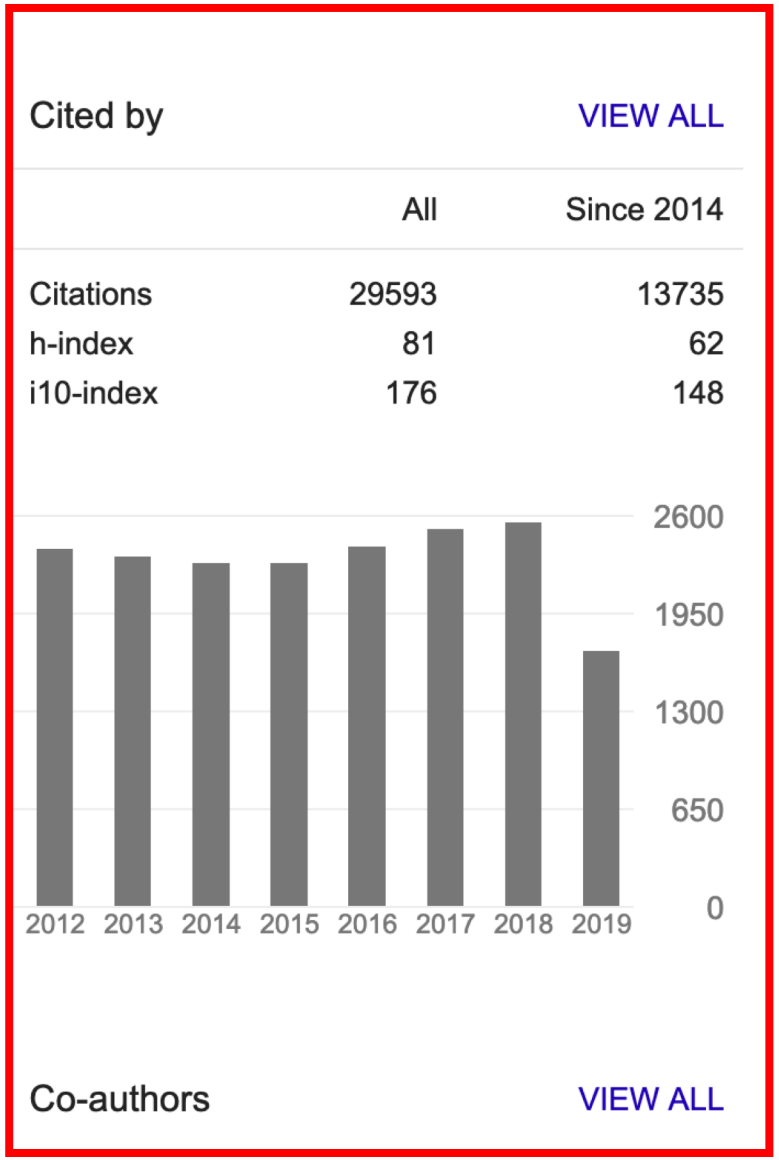
NCBI

PubMed.gov

US National Library of Medicine
National Institutes of Health

Google Scholar

JSTOR



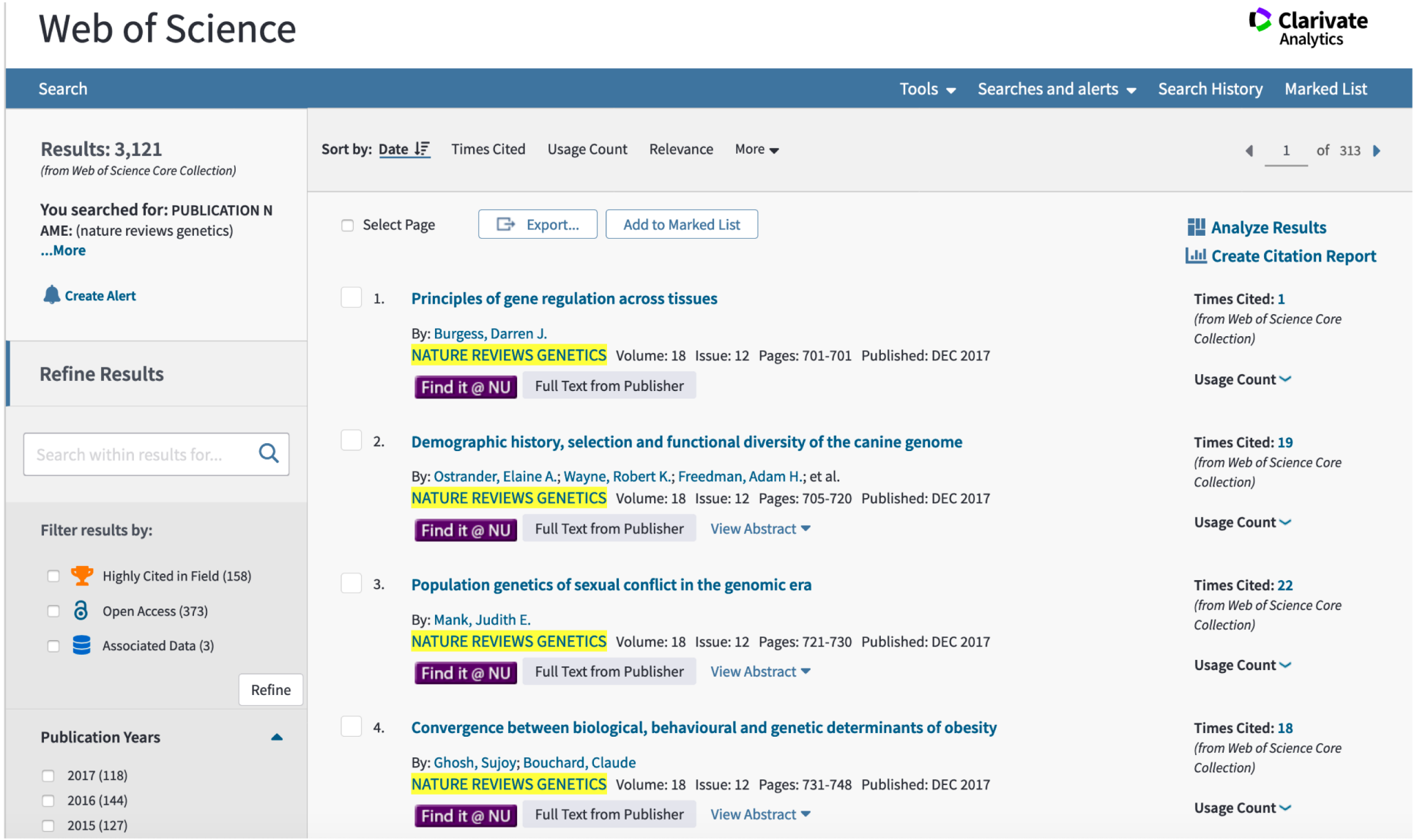
How many times will my article be cited?

…if it is published in <insert favorite journal here>

Project Goals

"So, how many times will my paper be cited?"

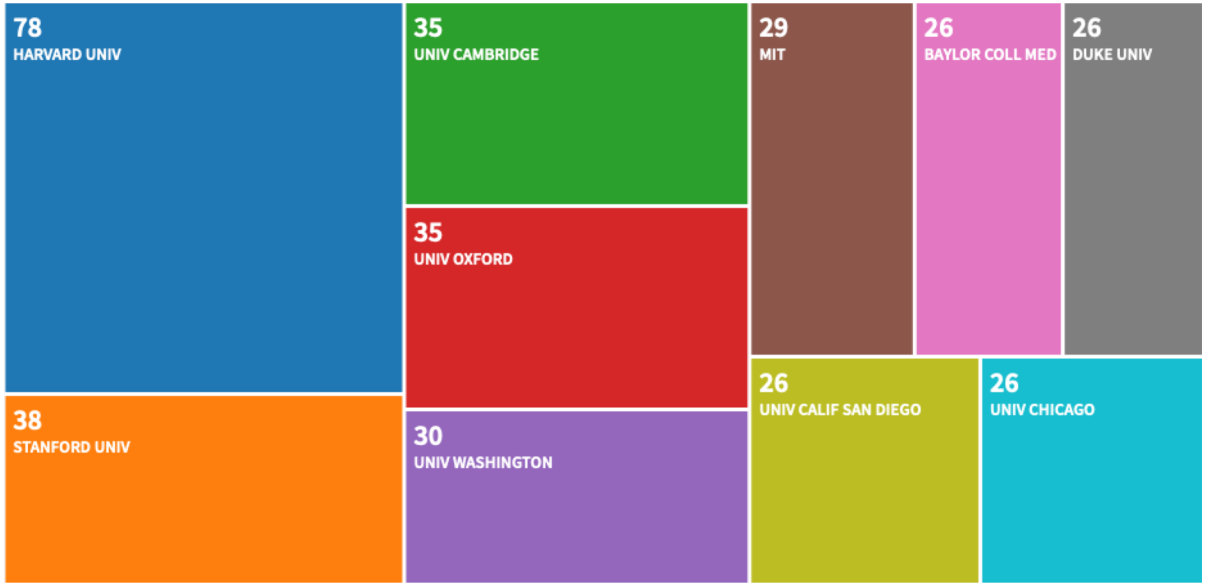
- Acquire citation metrics from web of science
  - 2000 – 2017
  - Nature Reviews Genetics
  - Title, authors, publication year, citation count (total and individual years)
- Visualization of citation metrics
- Use machine learning to predict citation counts for individual articles
  - Title of the article
  - Publication year
  - Name of senior author



Results can be downloaded as an .xls file
For our analyses, this was reformatted as a csv and cleaned up with pandas
Visualizations were generated with matplotlib

Summary (and fun facts)

- 3121 articles published between 2000 and 2017
- The highest cited paper was cited >5000 times
- On average, a NRG paper is cited 115 times
- Half of the dataset had no citations



Goal

To predict whether a paper has above- or below-median total citation count based on its:

Title
Authors

	Title	Authors	Publication Year	Total Citations	Average per Year	1997	1998	1999	2000	2001	...	2008	2009	2010	2011	2012
0	RNA-Seq: a revolutionary tool for transcriptomics	Wang, Zhong; Gerstein, Mark; Snyder, Michael	2009	5424	493.09	0	0	0	0	0	...	0	69	223	341	469
1	Network biology: Understanding the cell's func...	Barabasi, AL; Oltvai, ZN	2004	4202	262.63	0	0	0	0	0	...	248	326	285	316	359
2	The fundamental role of epigenetic events in c...	Jones, PA; Baylin, SB	2002	3855	214.17	0	0	0	0	0	...	262	263	284	292	270
3	APPLICATIONS OF NEXT-GENERATION SEQUENCING Seq...	Metzker, Michael L.	2010	3395	339.50	0	0	0	0	0	...	0	0	148	343	444
4	Microrna: Small RNAs with a big role in gene ...	He, L; Hannon, GJ	2004	3343	208.94	0	0	0	0	0	...	119	144	177	206	265

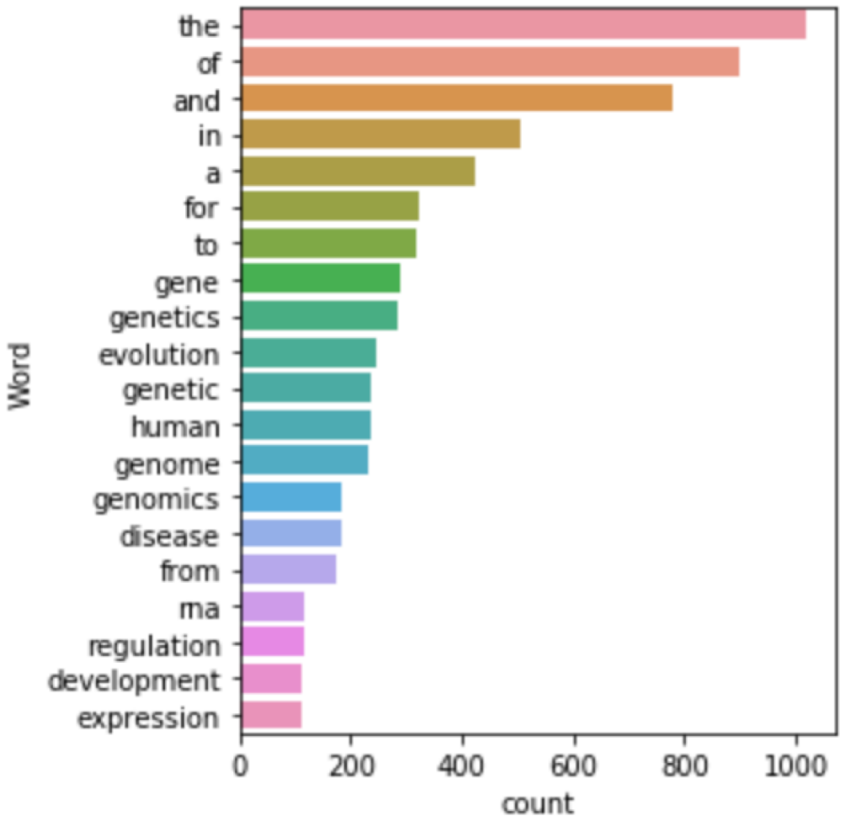
```
count     3121.000000
mean       115.244473
std        284.517875
min          0.000000
25%          0.000000
50%          2.000000
75%        130.000000
max       5424.000000
Name: Total Citations, dtype: float64
```

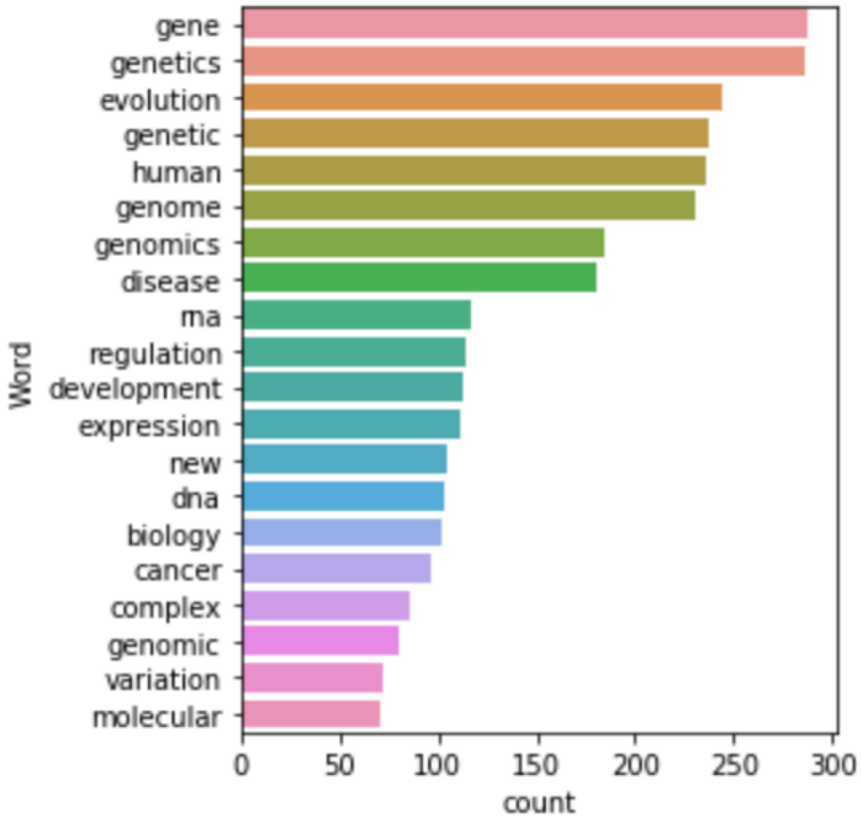
	finalcleaned_title	Average per Year	Total Citations	Category
0	rna seq revolutionary tool transcriptomics	493.09	5424	Top_half
1	network biology understanding cells functional...	262.63	4202	Top_half
2	fundamental role epigenetic events cancer	214.17	3855	Top_half
3	applications next generation sequencing sequen...	339.50	3395	Top_half
4	microrna small rnas big role gene regulation	208.94	3343	Top_half

Tokenization

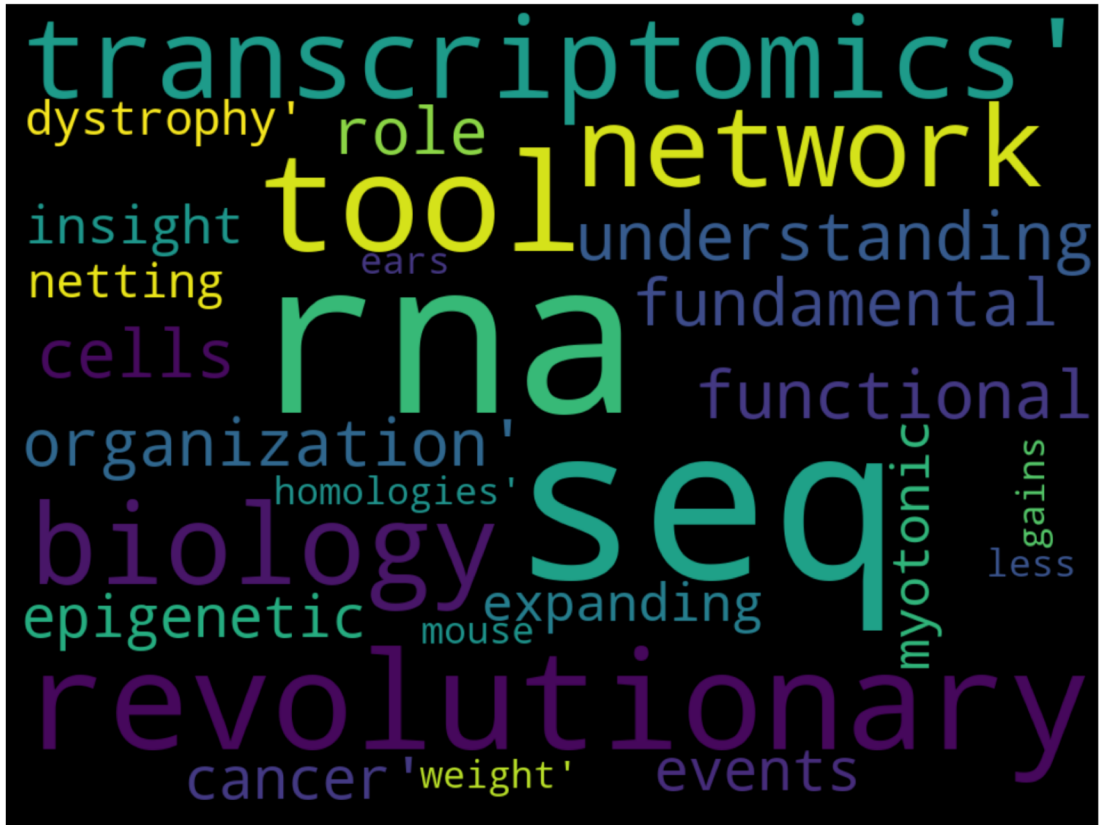
Title	cleaned_title	Authors	cleaned_authors
RNA-Seq: a revolutionary tool for transcriptomics	rna seq a revolutionary tool for transcriptomics	Wang, Zhong; Gerstein, Mark; Snyder, Michael	wang zhong gerstein mark snyder michael
Network biology: Understanding the cell's func...	network biology understanding the cells functi...	Barabasi, AL; Oltvai, ZN	barabasi al oltvai zn
The fundamental role of epigenetic events in c...	the fundamental role of epigenetic events in c...	Jones, PA; Baylin, SB	jones pa baylin sb
APPLICATIONS OF NEXT-GENERATION SEQUENCING Seq...	applications of next generation sequencing seq...	Metzker, Michael L.	metzker michael l
Micrornas: Small RNAs with a big role in gene ...	micrornas small rnas with a big role in gene r...	He, L; Hannon, GJ	he l hannon gj

Stopwords filtering

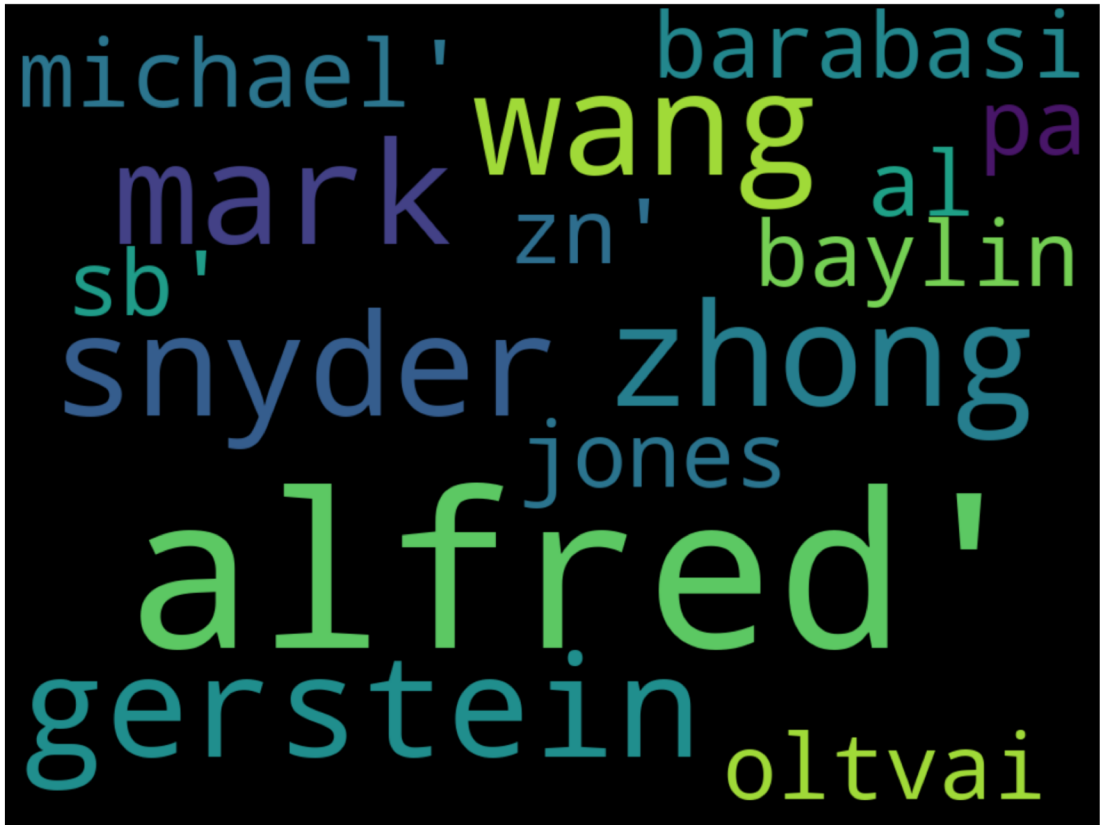




Most frequent words



Title



Author

TF-IDF vs. Count Vectorizer

- TF-IDF: weights down common words, gives more importance to rare words
- Count Vectorizer: count the number of words

```
from sklearn.feature_extraction.text import TfidfVectorizer
# initalise the vectoriser
tvec = TfidfVectorizer()
# fit the training data on the model
tvec.fit(X_train)

#transform training data into sparse matrix
X_train_tvec = tvec.transform(X_train)

# cross val score/ predict
tvec_score = cross_val_score(lr, X_train_tvec, y_train, cv=3)
```

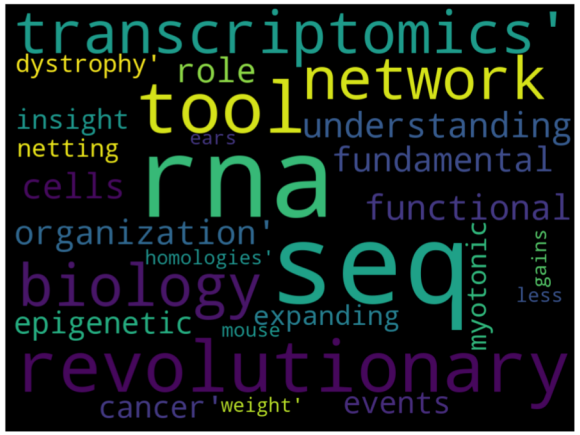
```
from sklearn.feature_extraction.text import CountVectorizer
# initalise the vectoriser
cvec = CountVectorizer()
# fit the training data on the model
cvec.fit(X_train)

#transform training data into sparse matrix
X_train_cvec = cvec.transform(X_train)

# cross val score/ predict
cvec_score = cross_val_score(lr, X_train_cvec, y_train, cv=3 )
X_train_cvec
```

ML: Predict citation based on title using logistic regression

```
TF-IDF Vectorizer Score: 0.714198009803197
Count Vectorizer Score: 0.7234898027045001
```

	params	scores
0	Count	0.723490
1	TF-IDF	0.714198

TF-IDF is used due to frequent appearance of words such as “RNA” and “Seq” in a genomics journal



Logistic Regression with TF-IDF

```
1  tfidf_vectorizer = TfidfVectorizer(max_df=0.8, max_features=1000)
2  xtrain, xval, ytrain, yval = train_test_split(traindf['finalcleaned_title'], traindf['Categ
3
4  # create TF-IDF features
5  xtrain_tfidf = tfidf_vectorizer.fit_transform(xtrain)
6  xval_tfidf = tfidf_vectorizer.transform(xval)
7
8  from sklearn.linear_model import LogisticRegression
9  classifier = LogisticRegression()
10 classifier.fit(xtrain_tfidf, ytrain)
11 print(f"Training Data Score: {classifier.score(xtrain_tfidf, ytrain)}")
12 print(f"Testing Data Score: {classifier.score(xval_tfidf, yval)}")
```

```
Training Data Score: 0.8441506410256411
Testing Data Score: 0.7472
```

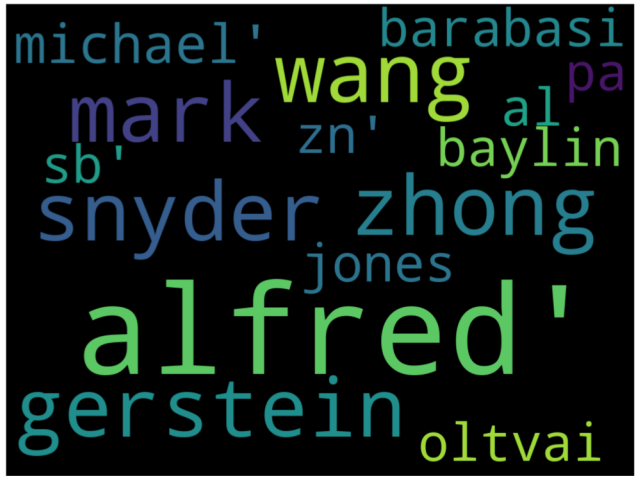
	Predictions	Actual
0	Top_half	Top_half
1	Top_half	Top_half
2	Bottom_half	Bottom_half
3	Top_half	Bottom_half
4	Bottom_half	Bottom_half
5	Bottom_half	Bottom_half
6	Bottom_half	Bottom_half
7	Top_half	Top_half
8	Top_half	Top_half
9	Top_half	Top_half

ML: Predict citation based on author using logistic regression

```
TF-IDF Vectorizer Score: 0.7767109411561862
Count Vectorizer Score: 0.8167510283504519
```

	params	scores
0	Count	0.816751
1	TF-IDF	0.776711

- Count vectorizer is used because we don't want to discard anyone's name…



Logistic Regression with Count Vectorizer

```
from sklearn.feature_extraction.text import CountVectorizer

xtrain, xval, ytrain, yval = train_test_split(traindf['finalcleaned_authors'], traindf['Cat

# initalise the vectoriser
cvec = CountVectorizer()

# fit the training data on the model, transform training data into sparse matrix
xtrain_cvec = cvec.fit_transform(xtrain)

from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
classifier.fit(xtrain_cvec, ytrain)


cvec_score = cross_val_score(lr, xtrain_cvec, ytrain, cv=10)
#print(cvec_score)
df_cvec = pd.DataFrame(xtrain_cvec.toarray(),columns=cvec.get_feature_names())
print(df_cvec.head())

cvec_score = cross_val_score(lr, X_train_cvec, y_train, cv=10)
print('Score:',tvec_score.mean())

```

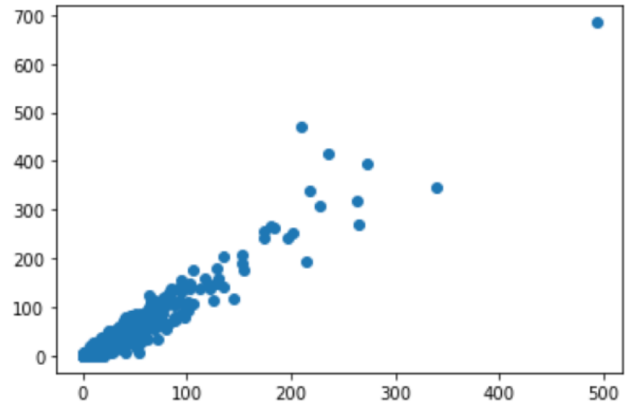
```
   aa  aaron  aarron  abbasi  abd  abdallah  abecasis  abel  abiola  abrahams  \
0   0      0       0       0    0         0         0     0       0         0
1   0      0       0       0    0         0         0     0       0         0
2   0      0       0       0    0         0         0     0       0         0
3   0      0       0       0    0         0         0     0       0         0
4   0      0       0       0    0         0         0     0       0         0

    ...    zollner  zon  zondervan  zoran  zou  zoya  zuben  zuber  zuccala  \
0   ...          0    0          0      0    0     0      0      0        0
1   ...          0    0          0      0    0     0      0      0        0
2   ...          0    0          0      0    0     0      0      0        0
3   ...          0    0          0      0    0     0      0      0        0
4   ...          0    0          0      0    0     0      0      0        0

   zuniga
0       0
1       0
2       0
3       0
4       0

[5 rows x 3420 columns]
Score: 0.7767109411561862
```

ML: Predict total citation based on previous trends using linear regression

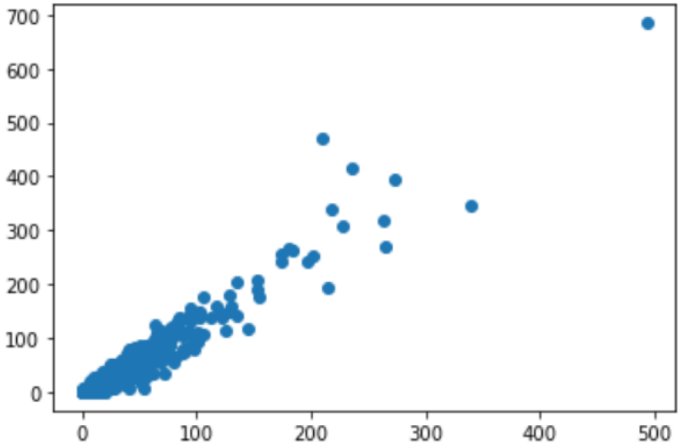
Average citation per year and last year's citations

```
1  peryear = naturegenetics["Average per Year"]
2  lastyear = naturegenetics["2017"]
3  plt.scatter(peryear, lastyear)
```

```
<matplotlib.collections.PathCollection at 0x11a77dd68>
```



Reshape




```
1  # Assign the data to X and y
2  # Note: Sklearn requires a two-dimensional array of values
3  # so we use reshape to create this
4
5  X = peryear.values.reshape(-1, 1)
6  y = lastyear.values.reshape(-1, 1)
7
8  print("Shape: ", X.shape, y.shape)
```

```
Shape:  (3121, 1) (3121, 1)
```

ML: Predict total citation based on previous trends using linear regression

```
# Use sklearn's `train_test_split` to split the data into training and testing

from sklearn.model_selection import train_test_split

# YOUR CODE HERE
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42)

```

```
# Create the model

# YOUR CODE HERE
from sklearn.datasets import make_regression
from sklearn.linear_model import LinearRegression
model = LinearRegression()

```

```
# Fit the model to the training data.

# YOUR CODE HERE
model.fit(X_train, y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
# Calculate the mean_squared_error and the r-squared value
# for the testing data

from sklearn.metrics import mean_squared_error, r2_score

# YOUR CODE HERE
# Use our model to predict a value
predicted = model.predict(X_test)

# Score the prediction with mse and r2
mse = mean_squared_error(y_test, predicted)
r2 = r2_score(y_test, predicted)

print(f"Mean Squared Error (MSE): {mse}")
print(f"R-squared (R2 ): {r2}")
```

```
Mean Squared Error (MSE): 66.3033064342903
R-squared (R2 ): 0.9544251088797852
```

```
# Call the `score` method on the model to show the r2 score

# YOUR CODE HERE
model.score(X_test, y_test)
```

```
0.954425108879785
```

```
totalyear = naturegenetics["Average per Year"]
lastyear2 = naturegenetics["2017"]
plt.scatter(totalyear, lastyear2)
```

```
<matplotlib.collections.PathCollection at 0x11ee43da0>
```