

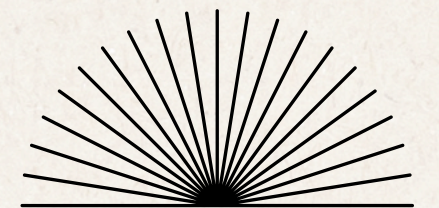
CMSC320: Introduction to Data Science

28 April 2025



PLANET TERP ANALYSIS

Mahati Gorthy



Introduction

01	Overview
02	Key Questions
03	Data
04	Initial Data Analysis
05	Machine Learning Techniques
06	Comparisons
07	Conclusion

Overview

- PlanetTerp is a student-driven platform at the University of Maryland that brings together:
 - **Professor reviews**
 - **Grade distributions**
 - **Course experiences**
- Ratings are **subjective**, but grades and student comments are **objective**.
- Predicting a professor's rating without using the actual student ratings allows us to:
 - Rely on **patterns** hidden in grades and review language
 - Understand how **perceptions** of difficulty, fairness, and teaching quality emerge from measurable outcomes
- It also raises a bigger question:
 - Is "**good teaching**" about high grades, positive feelings, or something else?
 - Modeling and analyzing this, could give insights into what students really value.

Key Questions

- How does the percentage of A grades **relate** to professor ratings?
- Is there a **relationship** between withdrawal rates and professor ratings?
- Can we **predict** a professor's rating based on non-rating features alone?

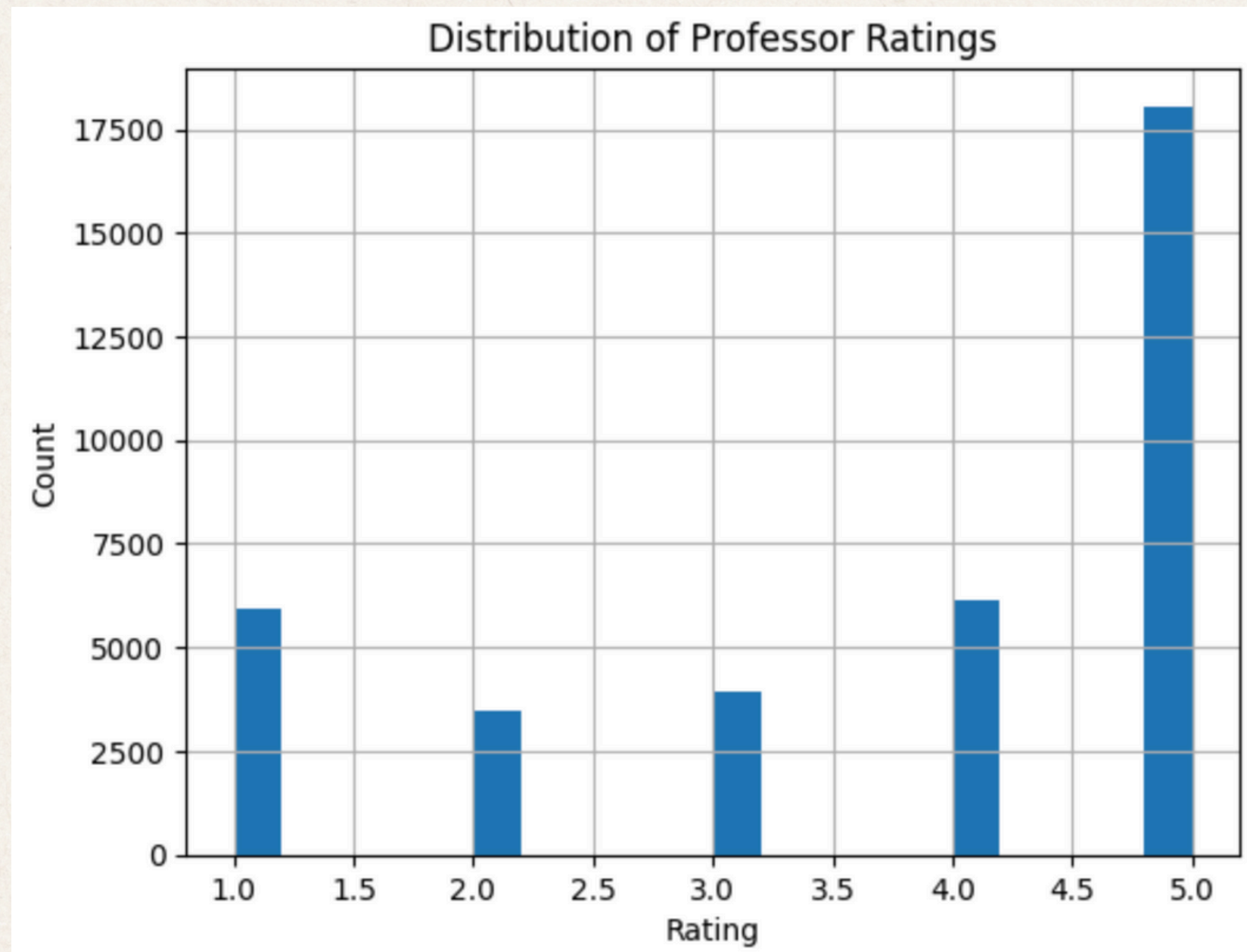
Data

- Used the PlanetTerp API to collect data on University of Maryland professors
- For each professor:
 - Retrieved basic info (name, ID)
 - Pulled all student reviews
 - Collected grade distributions (counts of A+, A, A-, etc.)
 - Pulled average rating (star rating) --- for supervised model training
 - Number of professors: 13427
 - num of student reviews: 37553
- Used Number requests library to call API endpoints
 - Parsed JSON responses to extract relevant fields
 - Built a Pandas DataFrame

name	slug	reviews	rating	A+	A	A-	B+	B	B-	C+	C	C-	D+	D	D-	F	W	Other
A Seyed	seyed	[]	NaN	18	9	0	1	0	0	0	0	0	0	0	0	0	0	0
Aaron Goldman	goldman_aaron	[]	NaN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Aaron Posner	posner	[]	NaN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Abani Pradhan	pradhan	[Pretty funny guy and he knows his stuff too. ...	3.6667	0	10	0	0	3	0	0	0	0	0	0	0	0	1	0
Abbey Morgan	morgan_abbey	[]	NaN	1	4	4	3	6	0	3	0	0	0	0	0	0	0	0
Abdel-Razak Kadry	kadry	[]	NaN	5	0	0	3	0	1	0	0	0	0	0	0	0	0	1
Abhijit Dasgupta	dasgupta_abhijit	[I took the ENAE version of this class but it ...	4.2500	4	5	4	2	2	0	0	0	0	0	0	0	0	1	1
Abigail Bickford	bickford	[]	NaN	15	44	2	1	1	0	0	0	0	0	0	0	0	1	0
Abigail McEwen	mcewen	[I was actually really excited to take this co...	2.6364	0	5	4	3	2	5	1	1	2	1	0	0	2	1	0
Abigail Romirowsky	romirowsky	[]	NaN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
Abigail Trozenski	trozenski	[]	NaN	3	2	3	2	2	0	1	0	0	0	1	0	0	3	1
Abolhassan Jawahery	jawahery	[Phys 405 is difficult. It just is. Now that w...	4.4000	1	1	2	1	3	1	0	0	0	0	0	0	0	0	1
Abram Kagan	kagan	[Guy knows his stuff, but that's about all tha...	2.7000	0	4	1	0	2	2	0	0	0	0	0	0	0	0	0
Adam Beissel	beissel	[]	NaN	0	1	3	6	11	7	0	7	4	1	1	0	7	8	0
Adam Binkley	binkley	[This teacher is great for english 101, a clas...	5.0000	0	6	2	3	2	1	0	0	1	0	3	0	1	1	0

Initial Data Analysis

Distribution of Professor Ratings

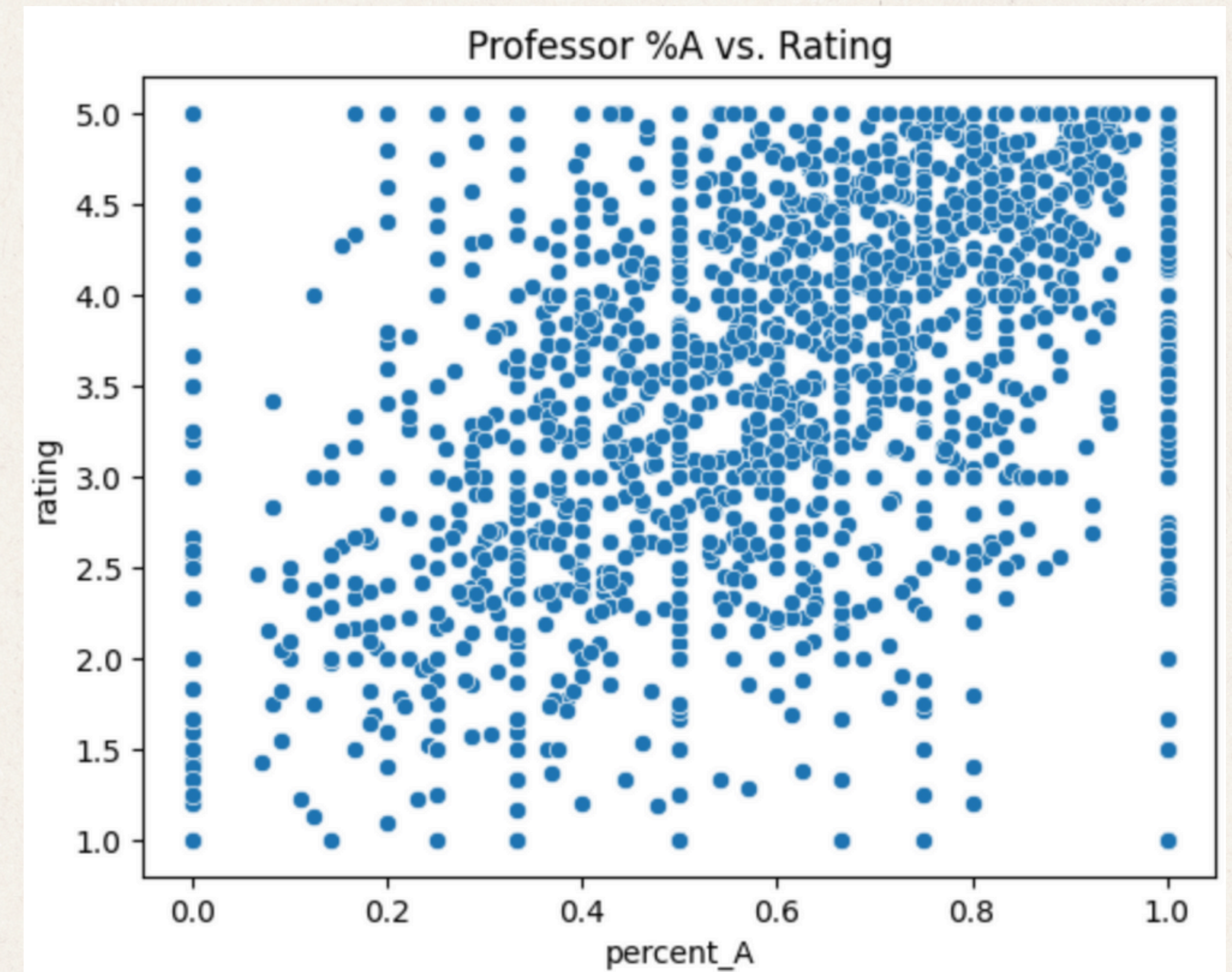


- Most professor ratings cluster around 2.5–4.5, with a slight skew toward higher ratings.
- A few professors received very low (around 1.0) while a large number of professors received a high (5.0) ratings.

Initial Data Analysis

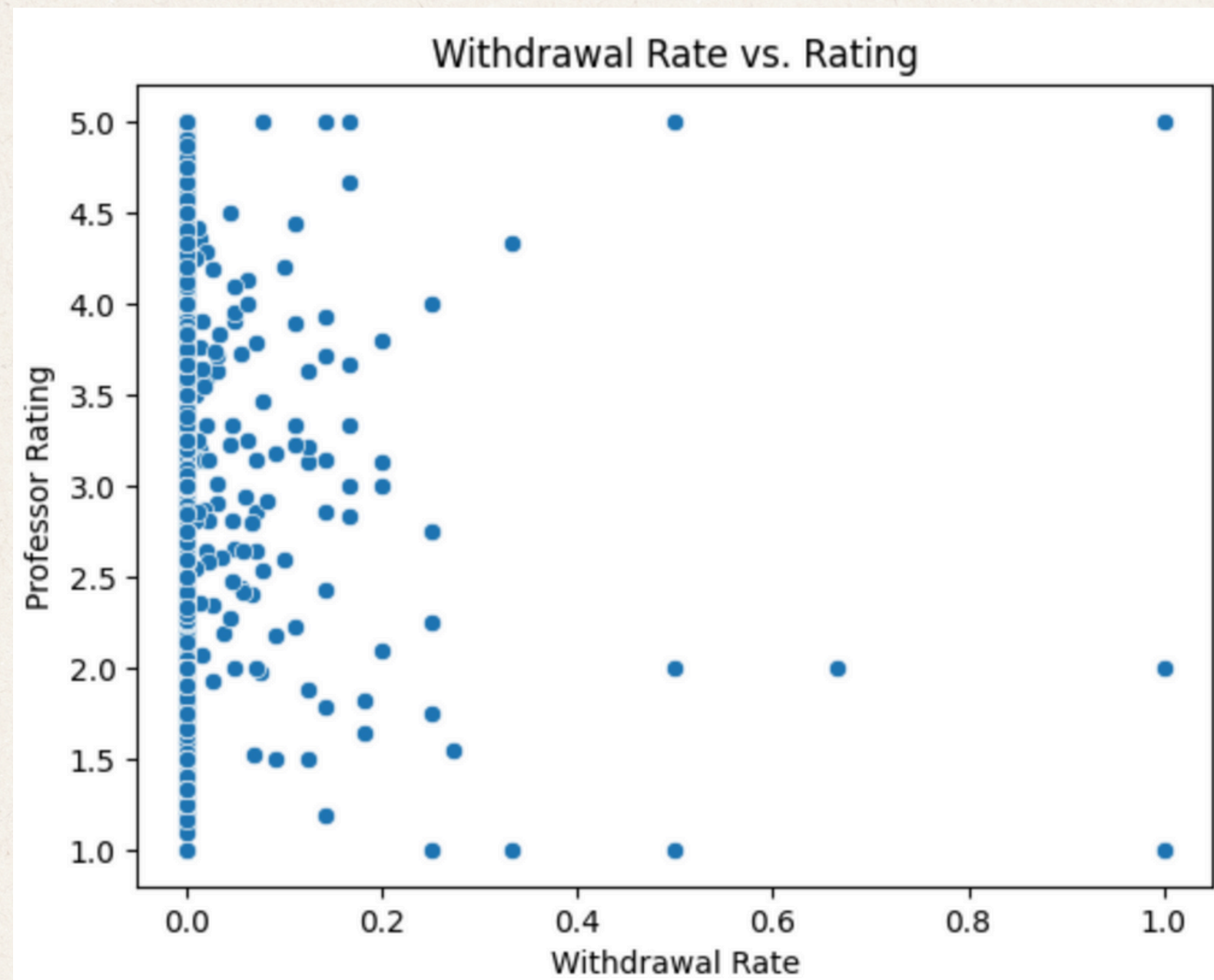
% of A Grades vs. Rating

- No clear linear relationship between the percentage of A grades and professor ratings.
- Some professors with a lower % of A's still have high ratings, and vice versa.



Initial Data Analysis

Withdrawal Rate vs. Rating



- Most professors have low withdrawal rates.
- A few professors with higher withdrawal rates still maintain good ratings.

Machine Learning Techniques

Model	Description
Linear Regression	Predicts ratings as a straight weighted combination of features (with no other non-linear effects).
Random Forest Regression	A group of decision trees working together to find complex patterns in the data
Support Vector Regression	Transforms input features into a higher-dimensional space to better fit a function, making it good for solving non-linear problems.

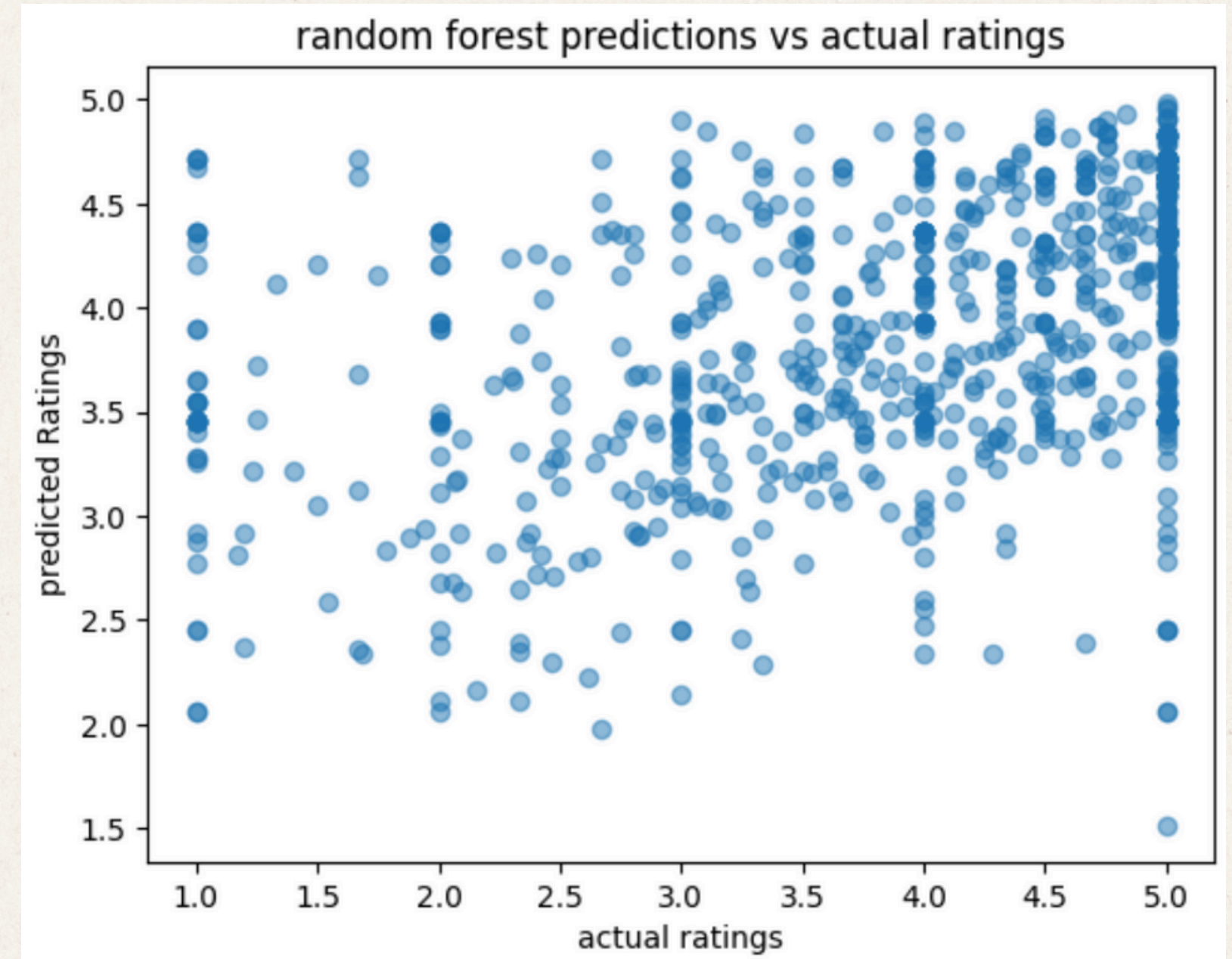
Machine Learning Techniques

Model	RMSE	R ²
Linear Regression	1.0836	0.0921
Random Forest	1.0108	0.2099
Support Vector Regression	1.0734	0.1091
Ridge Regression (an additional analysis that I completed)	1.0185	0.1979

Machine Learning Techniques

Random Forest: Best Model

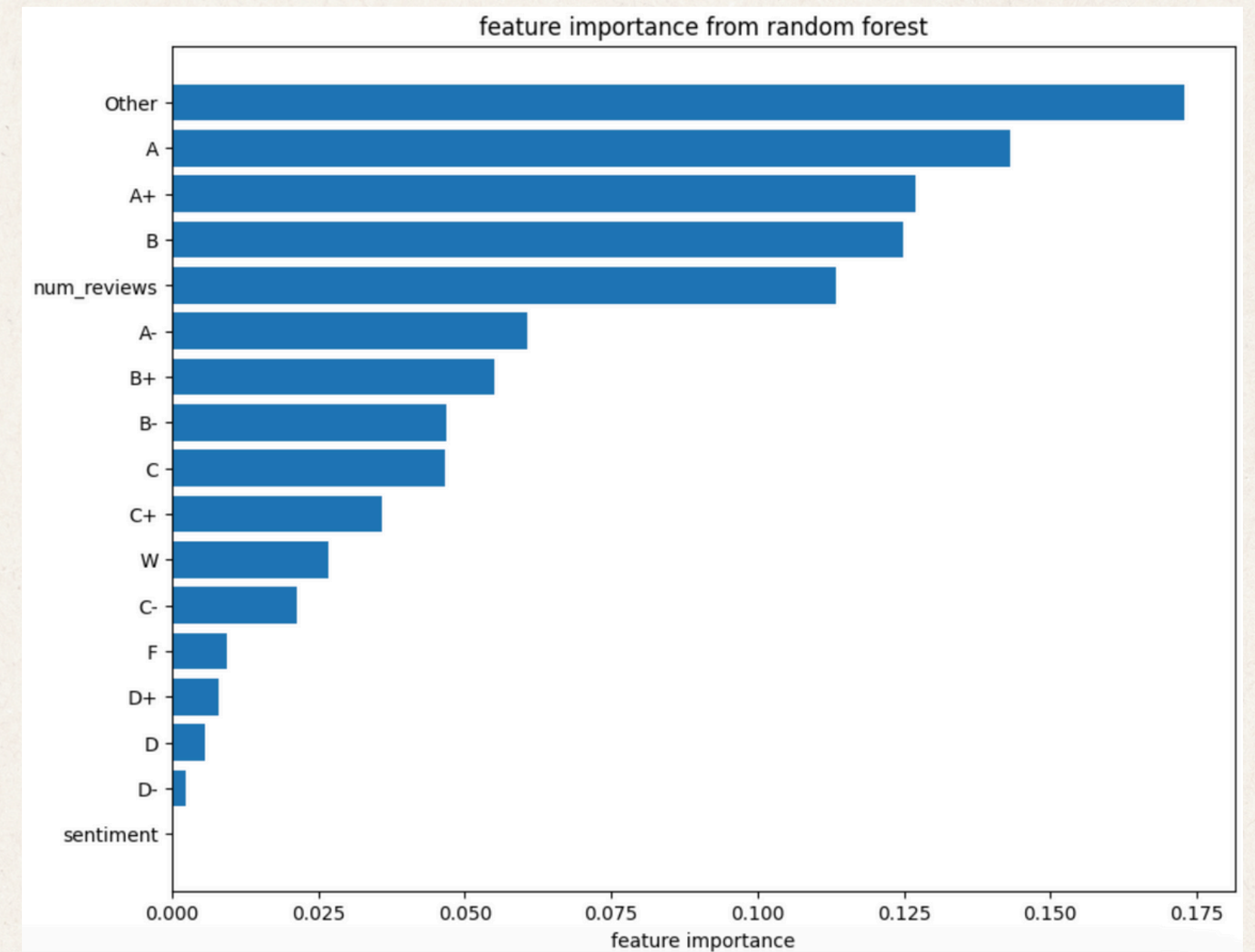
- Lowest RMSE (1.0108) and almost highest R^2 (0.2099) among the models.
- Captures non-linear patterns between features (sentiment, grades, withdrawals) and ratings.



Machine Learning Techniques

Feature Importance (for Random Forest)

- Top Feature: Number of "Other" grades is the most influential in predicting professor ratings.
- High Importance: Counts of A, A+, and B grades also significantly contribute to the model.
- Lower Importance: Sentiment score and number of reviews contribute minimally to the prediction.



Comparisons

Linear Regression

- Method: Assumes a simple, linear relationship between features (sentiment, grades, withdrawal rate, etc.) and professor rating.
- Performance:
 - RMSE: 1.0836 (highest among all models)
 - R^2 : 0.0921 (very poor fit)
- Takeaway:
 - Linear regression seems to be too simple for this task. It doesn't capture the more complex relationships in the data.

Support Vector Regression (SVR)

- Method: Attempts to fit the data within a higher-dimensional feature space, and can model non-linear relationships.
- Performance:
 - RMSE: 1.0734
 - R^2 : 0.1091
- Takeaway:
 - SVR performed better than linear regression but still not too great. It could not model the complex structure effectively. Even with hyperparameter tuning (like the kernel choice, epsilon value, etc.), it did not perform significantly better.

Comparisons

Random Forest Regression

- Method: An ensemble of decision trees that averages many models and a voting system to reduce variance and capture more non-linear ideas.
- Performance:
 - RMSE: 1.0108 (lowest among models)
 - R^2 : 0.2099 (positive, explaining ~20% of variance)
- Takeaway:
 - Random Forest was the most successful model. It was able to capture complex, non-linear relationships without much tuning. It handles feature interactions, missing values, and different feature scales well.

Ridge Regression

- Method: Linear regression with L2 regularization, using engineered features (sentiment, leniency score, hardness index, withdrawal rate).
- Performance:
 - RMSE: 1.0185
 - R^2 : 0.1979
- Takeaway:
 - Ridge regression improved slightly compared to plain linear regression due to better feature engineering and regularization.
 - However, its performance was still worse than Random Forest. It indicates that linear combinations, even regularized, are not enough for this task.
 - Feature Coefficients:
 - Leniency score: +1.34, Hardness index: -1.20, Withdrawal rate: -1.54, Sentiment: 0.00
 - → Sentiment was not influential in this model; withdrawal rate had a strong negative effect.

Overall Conclusion

- Random Forest outperformed other models in terms of RMSE.
- Simpler models like Linear and Ridge regression could not capture the non-linear, complex nature of the relationship between review sentiment, grades, and professor ratings.

Conculsions

Trained and evaluated four models:

- Linear Regression, Support Vector Regression (SVR), Random Forest Regression, and Ridge Regression with Random Forest Regression as the most successful, with RMSE of 1.0108 and R^2 score of 0.2099

Evaluation Method

- Models were evaluated using:
 - Root Mean Squared Error (RMSE): Lower values mean better predictive accuracy as this value quantifies the average magnitude of the differences between predicted and actual values in a dataset.
 - R^2 Score: Measures the proportion of variance explained; closer to 1 is better, as it is the proportion of the variation in the dependent variable that is predictable from the independent variable.
- Evaluation was done on a 20% test set (train-test split).

Conculsions

Most Important Features:

From the Random Forest model's feature importance:

- Sentiment of student reviews was the most influential feature by a large margin.
- Number of reviews and percentage of A grades were moderately important.
- Grade distributions (B's, C's, D's) and withdrawal rates had smaller impacts.

This shows that how students feel about a professor matters more than the grades they receive.

Final Takeaways

- The evaluation is strong because:
 - The appropriate regression metrics (RMSE and R^2).
 - Evaluated on unseen data (test set) to properly measure model performance.
 - Interpreted feature importance to understand what factors influence ratings.

Thank you

```
# from itertools import tee
def get_sections(text_line):
```

```
if test_size is None or (test_size>test_size_val):
    test_size = test_size_val
else:
    test_size = test_size

# Split the data into training and testing sets
train_data, test_data = train_test_split(X, y, test_size=test_size, random_state=42)

# Create a Gradient Boosting Regressor
model = GradientBoostingRegressor()

# Fit the model to the training data
model.fit(train_data, train_data['y'])

# Predict the target values for the test data
y_pred = model.predict(test_data)

# Calculate the Mean Squared Error (MSE)
mse = mean_squared_error(test_data['y'], y_pred)

# Print the MSE
print('MSE: {}'.format(mse))
```

	feature	average_rating	type	reviewer	name	rating	reviewer_id	is_verified	is_top_reviewer	is_verified_top_reviewer	is_verified_top_reviewer_id	is_verified_top_reviewer_id_val	reviewer_rating	reviewer_rating_val
0	product	4.5	product	1	Apple	5	1	1	1	1	1	1	5	5
1	product	4.5	product	2	Apple	5	2	1	1	1	1	1	5	5
2	product	4.5	product	3	Apple	5	3	1	1	1	1	1	5	5
3	product	4.5	product	4	Apple	5	4	1	1	1	1	1	5	5
4	product	4.5	product	5	Apple	5	5	1	1	1	1	1	5	5
5	product	4.5	product	6	Apple	5	6	1	1	1	1	1	5	5
6	product	4.5	product	7	Apple	5	7	1	1	1	1	1	5	5
7	product	4.5	product	8	Apple	5	8	1	1	1	1	1	5	5
8	product	4.5	product	9	Apple	5	9	1	1	1	1	1	5	5
9	product	4.5	product	10	Apple	5	10	1	1	1	1	1	5	5
10	product	4.5	product	11	Apple	5	11	1	1	1	1	1	5	5
11	product	4.5	product	12	Apple	5	12	1	1	1	1	1	5	5
12	product	4.5	product	13	Apple	5	13	1	1	1	1	1	5	5
13	product	4.5	product	14	Apple	5	14	1	1	1	1	1	5	5
14	product	4.5	product	15	Apple	5	15	1	1	1	1	1	5	5
15	product	4.5	product	16	Apple	5	16	1	1	1	1	1	5	5
16	product	4.5	product	17	Apple	5	17	1	1	1	1	1	5	5
17	product	4.5	product	18	Apple	5	18	1	1	1	1	1	5	5
18	product	4.5	product	19	Apple	5	19	1	1	1	1	1	5	5
19	product	4.5	product	20	Apple	5	20	1	1	1	1	1	5	5
20	product	4.5	product	21	Apple	5	21	1	1	1	1	1	5	5
21	product	4.5	product	22	Apple	5	22	1	1	1	1	1	5	5
22	product	4.5	product	23	Apple	5	23	1	1	1	1	1	5	5
23	product	4.5	product	24	Apple	5	24	1	1	1	1	1	5	5
24	product	4.5	product	25	Apple	5	25	1	1	1	1	1	5	5
25	product	4.5	product	26	Apple	5	26	1	1	1	1	1	5	5
26	product	4.5	product	27	Apple	5	27	1	1	1	1	1	5	5
27	product	4.5	product	28	Apple	5	28	1	1	1	1	1	5	5
28	product	4.5	product	29	Apple	5	29	1	1	1	1	1	5	5
29	product	4.5	product	30	Apple	5	30	1	1	1	1	1	5	5
30	product	4.5	product	31	Apple	5	31	1	1	1	1	1	5	5
31	product	4.5	product	32	Apple	5	32	1	1	1	1	1	5	5
32	product	4.5	product	33	Apple	5	33	1	1	1	1	1	5	5
33	product	4.5	product	34	Apple	5	34	1	1	1	1	1	5	5
34	product	4.5	product	35	Apple	5	35	1	1	1	1	1	5	5
35	product	4.5	product	36	Apple	5	36	1	1	1	1	1	5	5
36	product	4.5	product	37	Apple	5	37	1	1	1	1	1	5	5
37	product	4.5	product	38	Apple	5	38	1	1	1	1	1	5	5
38	product	4.5	product	39	Apple	5	39	1	1	1	1	1	5	5
39	product	4.5	product	40	Apple	5	40	1	1	1	1	1	5	5
40	product	4.5	product	41	Apple	5	41	1	1	1	1	1	5	5
41	product	4.5	product	42	Apple	5	42	1	1	1	1	1	5	5
42	product	4.5	product	43	Apple	5	43	1	1	1	1	1	5	5
43	product	4.5	product	44	Apple	5	44	1	1	1	1	1	5	5
44	product	4.5	product	45	Apple	5	45	1	1	1	1	1	5	5
45	product	4.5	product	46	Apple	5	46	1	1	1	1	1	5	5
46	product	4.5	product	47	Apple	5	47	1	1	1	1	1	5	5
47	product	4.5	product	48	Apple	5	48	1	1	1	1	1	5	5
48	product	4.5	product	49	Apple	5	49	1	1	1	1	1	5	5
49	product	4.5	product	50	Apple	5	50	1	1	1	1	1	5	5
50	product	4.5	product	51	Apple	5	51	1	1	1	1	1	5	5
51	product	4.5	product	52	Apple	5	52	1	1	1	1	1	5	5
52	product	4.5	product	53	Apple	5	53	1	1	1	1	1	5	5
53	product	4.5	product	54	Apple	5	54	1	1	1	1	1	5	5
54	product	4.5	product	55	Apple	5	55	1	1	1	1	1	5	5
55	product	4.5	product	56	Apple	5	56	1	1	1	1	1	5	5
56	product	4.5	product	57	Apple	5	57	1	1	1	1	1	5	5
57	product	4.5	product	58	Apple	5	58	1	1	1	1	1	5	5
58	product	4.5	product	59	Apple	5	59	1	1	1	1	1	5	5
59	product	4.5	product	60	Apple	5	60	1	1	1	1	1	5	5
60	product	4.5	product	61	Apple	5	61	1	1	1	1	1	5	5
61	product	4.5	product	62	Apple	5	62	1	1	1	1	1	5	5
62	product	4.5	product	63	Apple	5	63	1	1	1	1	1	5	5
63	product	4.5	product	64	Apple	5	64	1	1	1	1	1	5	5
64	product	4.5	product	65	Apple	5	65	1	1	1	1	1	5	5
65	product	4.5	product	66	Apple	5	66	1	1	1	1	1	5	5
66	product	4.5	product	67	Apple	5	67	1	1	1	1	1	5	5
67	product	4.5	product	68	Apple	5	68	1	1	1	1	1	5	5
68	product	4.5	product	69	Apple	5	69	1	1	1	1	1	5	5
69	product	4.5	product	70	Apple	5	70	1	1	1	1	1	5	5
70	product	4.5	product	71	Apple	5	71	1	1	1	1	1	5	5
71	product	4.5	product	72	Apple	5	72	1	1	1	1	1	5	5
72	product	4.5	product	73	Apple	5	73	1	1	1	1	1	5	5
73	product	4.5	product	74	Apple	5	74	1	1	1	1	1	5	5
74	product	4.5	product	75	Apple	5	75	1	1	1	1	1	5	5
75	product	4.5	product	76	Apple	5	76	1	1	1	1	1	5	5
76	product	4.5	product	77	Apple	5	77	1	1	1	1	1	5	5
77	product	4.5	product	78	Apple	5	78	1	1	1	1	1	5	5
78	product	4.5	product	79	Apple	5	79	1	1	1	1	1	5	5
79	product	4.5	product	80	Apple	5	80	1	1	1	1	1	5	5
80	product	4.5	product	81	Apple	5	81	1	1	1	1	1	5	5
81	product	4.5	product	82	Apple	5	82	1	1	1	1	1	5	5
82	product	4.5	product	83	Apple	5	83	1	1	1	1	1	5	5
83	product	4.5	product	84	Apple	5	84	1	1	1	1	1	5	5
84	product	4.5	product	85	Apple	5	85	1	1	1	1	1	5	5
85	product	4.5	product	86	Apple	5	86	1	1	1	1	1	5	5
86	product	4.5	product	87	Apple	5	87	1	1	1	1	1	5	5
87	product	4.5	product	88	Apple	5	88	1	1	1	1	1	5	5
88	product	4.5	product	89	Apple	5	89	1	1	1	1	1	5	5
89	product	4.5	product	90	Apple	5	90	1	1	1	1	1	5	5
90	product	4.5	product	91	Apple	5	91	1	1	1	1	1	5	5
91	product	4.5	product	92	Apple	5	92	1	1	1	1	1	5	5
92	product	4.5	product	93	Apple	5	93	1	1	1	1	1	5	5
93	product	4.5	product	94	Apple	5	94	1	1	1	1	1	5	5
94	product	4.5	product	95	Apple	5	95	1	1	1	1	1	5	5
95	product	4.5	product	96	Apple	5	96	1	1	1	1	1	5	5
96	product	4.5	product	97	Apple	5	97	1	1	1	1	1	5	5
97	product	4.5	product	98	Apple	5	98	1	1	1	1	1	5	5
98	product	4.5	product	99	Apple	5	99	1	1	1	1	1	5	5
99	product	4.5	product	100	Apple	5	100	1	1	1	1	1	5	5

```
# Print the MSE
print('MSE: {}'.format(mse))

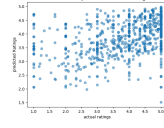
# Create a Gradient Boosting Regressor
model = GradientBoostingRegressor()

# Fit the model to the training data
model.fit(train_data, train_data['y'])

# Predict the target values for the test data
y_pred = model.predict(test_data)

# Calculate the Mean Squared Error (MSE)
mse = mean_squared_error(test_data['y'], y_pred)

# Print the MSE
print('MSE: {}'.format(mse))
```



```
# Print the MSE
print('MSE: {}'.format(mse))

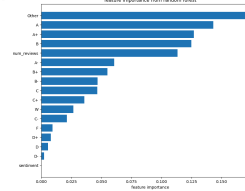
# Create a Gradient Boosting Regressor
model = GradientBoostingRegressor()

# Fit the model to the training data
model.fit(train_data, train_data['y'])

# Predict the target values for the test data
y_pred = model.predict(test_data)

# Calculate the Mean Squared Error (MSE)
mse = mean_squared_error(test_data['y'], y_pred)

# Print the MSE
print('MSE: {}'.format(mse))
```



```
# Print the MSE
print('MSE: {}'.format(mse))

# Create a Gradient Boosting Regressor
model = GradientBoostingRegressor()

# Fit the model to the training data
model.fit(train_data, train_data['y'])

# Predict the target values for the test data
y_pred = model.predict(test_data)

# Calculate the Mean Squared Error (MSE)
mse = mean_squared_error(test_data['y'], y_pred)

# Print the MSE
print('MSE: {}'.format(mse))
```