Predictive Modelling

Regularization

Jonathan Mwaura

Khoury College of Computer Science

July 16, 2024

The Roux Institute
at Northeastern University

Introduction

Textbook

Reading: Chapter 3 of: Gareth James et al (2021) . An Introduction to Statistical Learning (2nd Edition) .

`https://www.statlearning.com/`

Acknowledgements

These slides have been adapted from the following Professors:

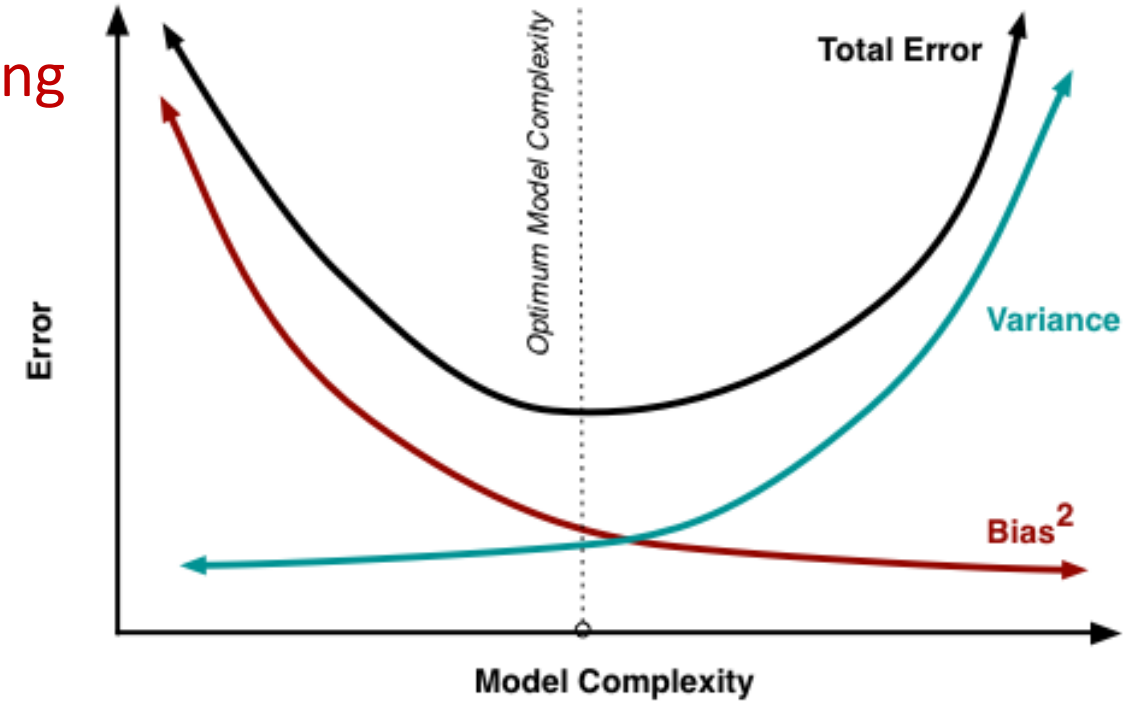
1) Andrew Ng - Stanford
2) Eric Eaton - UPenn
3) David Sontag - MIT
4) Alina Oprea - Northeastern

Outline

- Brief review
- Gradient descent
  - Batch algorithm
  - Line search optimization
- Gradient descent for linear regression
- Regularization
  - Ridge and Lasso regression
  - Gradient descent for ridge regression

Bias-Variance Tradeoff

Under-fitting



Over-fitting

- Bias = Difference between estimated and true models
- Variance = Model difference on different training sets

MSE is proportional to Bias + Variance

Regularization

- A method for automatically controlling the complexity of the learned hypothesis
- **Idea**: penalize for large values of θ_j
  - Can incorporate into the cost function
  - Works well when we have a lot of features, each that contributes a bit to predicting the label
- Can also address overfitting by eliminating features (either manually or via model selection)

Reduce model complexity

Reduce model variance

Ridge regression

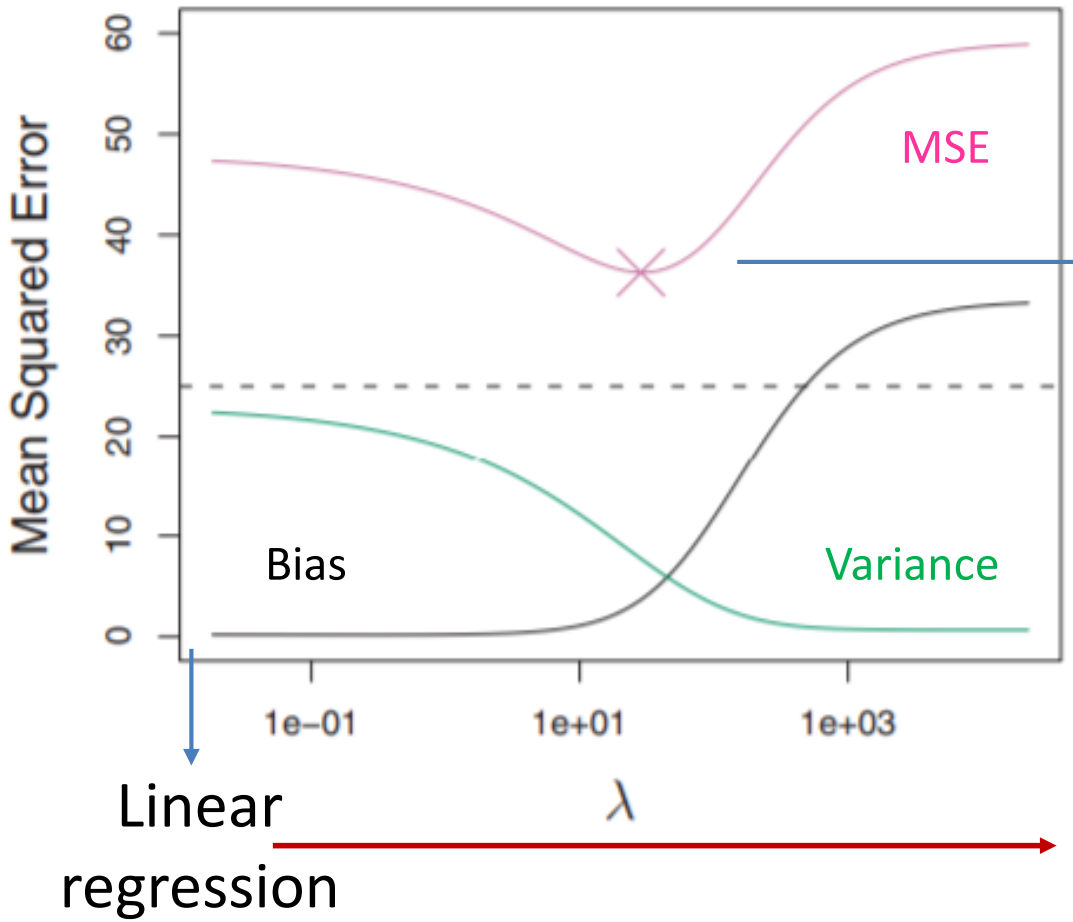
- Linear regression objective function

$$J(\theta) = \underbrace{\frac{1}{2}\sum_{i=1}^{N} \left(h_\theta(x_i) - y_i \right)^2}_{\text{model fit to data}} + \underbrace{\frac{\lambda}{2}\sum_{j=1}^{d} \theta_j^2}_{\text{regularization}}$$

  - λ is the regularization parameter ($\lambda \geq 0$)
  - No regularization on θ_0!

    - If λ = 0, we train linear regression
    - If λ is large, the coefficients will shrink close to 0

Bias-Variance Tradeoff

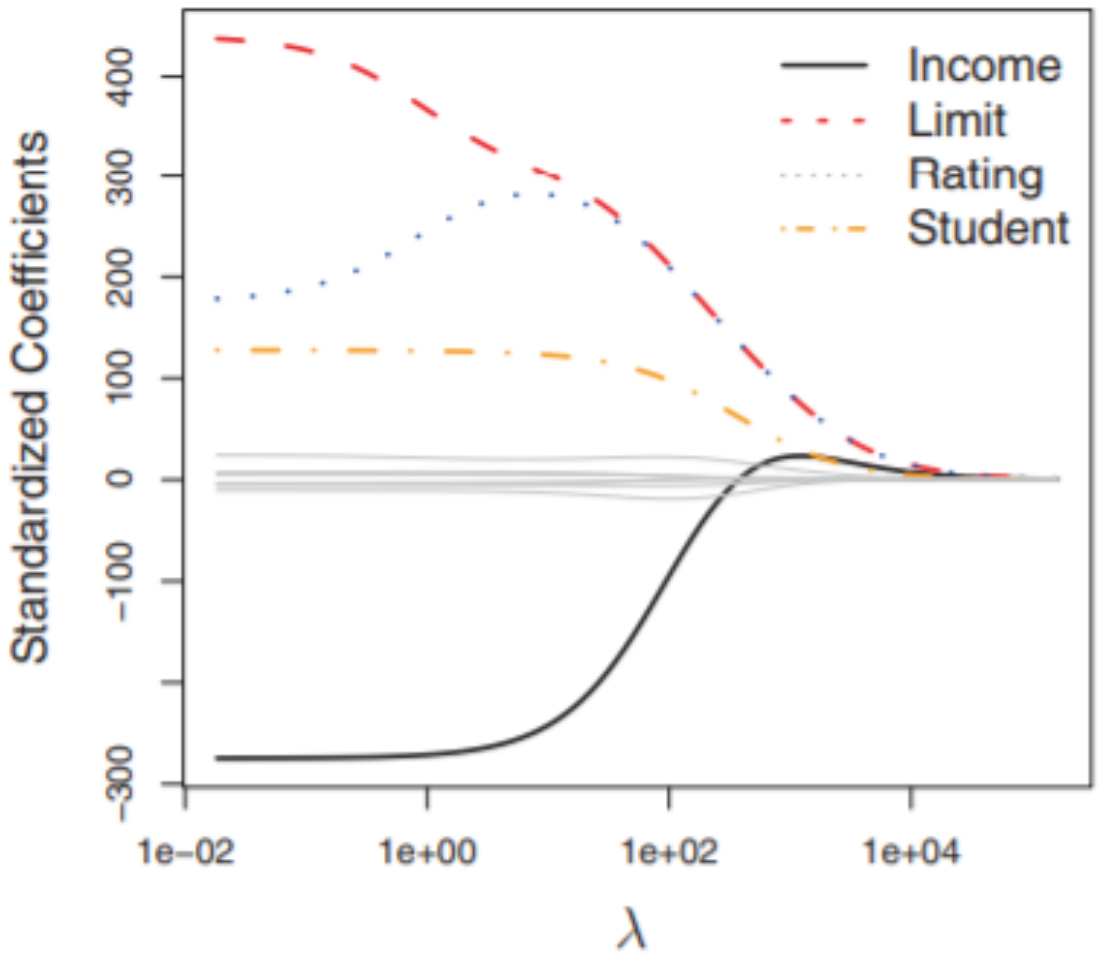


Optimal
Ridge regression

Linear
regression

Reduced model
complexity

Coefficient shrinkage



Predict credit card balance

GD for Ridge Regression

Min MSE

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{N} (h_\theta(x_i) - y_i)^2 + \frac{\lambda}{2}\sum_{j=1}^{d} \theta_i^2$$

Gradient update: $\theta_0 \leftarrow \theta_0 - \alpha \sum_{i=1}^{N}(h_\theta(x_i) - y_i)$

$$\theta_j \leftarrow \theta_j - \alpha \sum_{i=1}^{N}(h_\theta(x_i) - y_i)x_{ij} - \underbrace{\alpha\lambda\theta_j}_{\text{Regularization}}$$

$$\theta_j \leftarrow \theta_j(1 - \alpha\lambda) - \alpha \sum_{i=1}^{N}(h_\theta(x_i) - y_i)x_{ij}$$

Lab example

```
us_youtube = pd.read_csv('us_youtube_videos.csv')
```

```
us_youtube.head(15)
```

	views	likes	dislikes	comment_count
0	748374	57527	2966	15954
1	2418783	97185	6146	12703
2	3191434	146033	5339	8181
3	343168	10172	666	2146
4	2095731	132235	1989	17518
5	119180	9763	511	1434
6	2103417	15993	2445	1970
7	817732	23663	778	3432
8	826059	3543	119	340
9	256426	12654	1363	2368
10	81377	655	25	177
11	104578	1576	303	1279
12	687582	114188	1333	8371
13	544770	7848	1171	3981
14	207532	7473	246	2120

Lab example

```
us_youtube = pd.read_csv('us_youtube_videos.csv')
```

```
us_youtube.head(15)
```

Features min-max normalized

	views	likes	dislikes	published_in_morning	comment_count
0	0.003321	0.010247	0.001771	1	15954
1	0.010738	0.017312	0.003671	0	12703
2	0.014168	0.026013	0.003189	1	8181
3	0.001521	0.001812	0.000398	0	2146
4	0.009303	0.023555	0.001188	1	17518
5	0.000527	0.001739	0.000305	1	1434
6	0.009337	0.002849	0.001460	0	1970
7	0.003629	0.004215	0.000465	1	3432
8	0.003665	0.000631	0.000071	1	340
9	0.001136	0.002254	0.000814	1	2368
10	0.000359	0.000117	0.000015	0	177
11	0.000462	0.000281	0.000181	0	1279
12	0.003051	0.020340	0.000796	1	8371
13	0.002416	0.001398	0.000699	1	3981
14	0.000919	0.001331	0.000147	1	2120

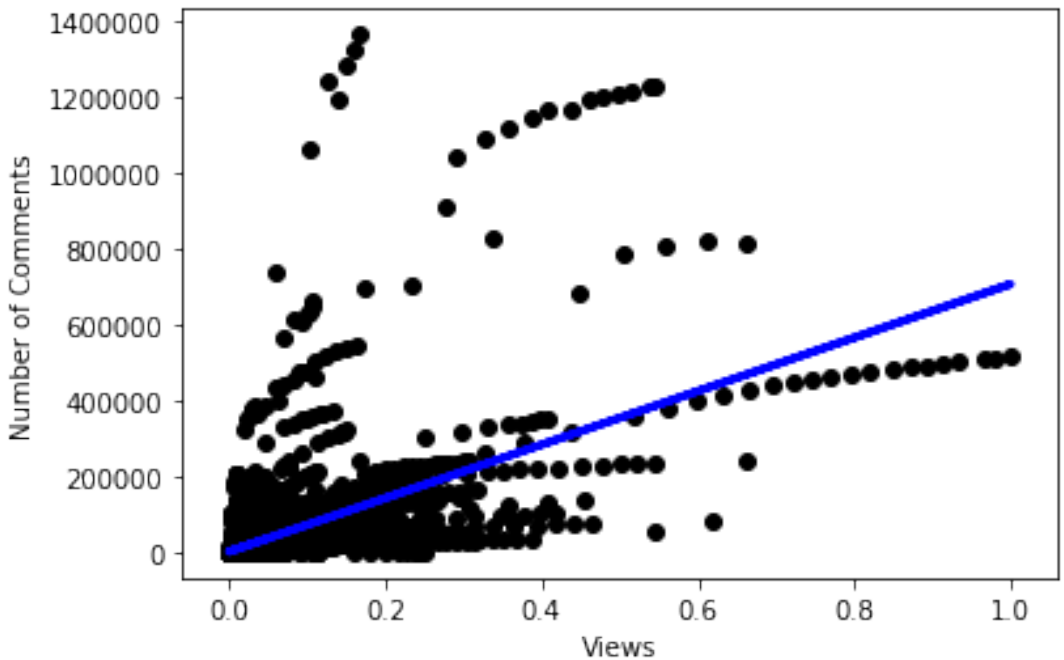
Simple LR

```
views_comments = us_youtube[['views', 'comment_count']]
```

```
reg = linear_model.LinearRegression()
reg.fit(views_comments.drop(columns='comment_count'), views_comments['comment_count'])

comments_pred = reg.predict(views_comments.drop(columns='comment_count'))

plt.scatter(views_comments['views'], views_comments['comment_count'], color='black')
plt.plot(views_comments['views'], comments_pred, color='blue', linewidth=3)
plt.xlabel('Views')
plt.ylabel('Number of Comments')
plt.show()
```



Simple LR

```
print("views coef: {}".format(reg.coef_[0]))
```

```
views coef: 704128.4873046515
```

```
mse_single = mean_squared_error(views_comments['comment_count'], comments_pred)
print("MSE: {}".format(mse_single))
print("RSE: {}".format(mse_single ** 0.5))
```

```
MSE: 866584512.2544774
RSE: 29437.807531378374
```

Without scaling response variable

```
views coef: 0.5171407389243756

MSE: 0.0004674386251650002
RSE: 0.021620328979111307
```

With scaling response variable

Residual Standard Error (RSE)

$$RSE = \sqrt{MSE}$$

Multiple LR

```
reg_multi = linear_model.LinearRegression()
reg_multi.fit(us_youtube.drop(columns='comment_count'), us_youtube['comment_count'])
```

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

```
comments_pred_multi = reg_multi.predict(us_youtube.drop(columns='comment_count'))
```

```
feature_names = us_youtube.drop(columns='comment_count').columns
for i in range(len(reg_multi.coef_)):
    print("{} coef: {}".format(feature_names[i], reg_multi.coef_[i]))
```

```
views coef: -441184.8341255368
likes coef: 845430.3290977236
dislikes coef: 1017058.2337598497
published_in_morning coef: -167.28387229563668
```

Without scaling response variable

```
mse_multi = mean_squared_error(us_youtube['comment_count'], comments_pred_multi)
print("MSE: {}".format(mse_multi))
print("RSE: {}".format(mse_multi ** 0.5))
```

```
MSE: 237564606.31275252
RSE: 15413.130970466465
```

```
views coef: -0.32402417347899976
likes coef: 0.6209185865668729
dislikes coef: 0.7469691342116144
published_in_morning coef: -0.0001228601127334361


MSE: 0.0001281431544094894
RSE: 0.011320033321924869
```

With scaling response variable
Lower MSE/RSE with multiple features

Review

- Gradient descent is a general optimization algorithm that can be applied in training
  - Minimize loss function (error metric)
- Gradient descent converges to local minima
- Requires selection of learning rate
- Bias-Variance tradeoff shows that both bias and variance need to be minimized
  - Regularization is a method to reduce model complexity and avoid overfitting
  - Ridge and Lasso regularization can be added to any loss function
  - Regularized model trained with Gradient Descent