LUTHER PREDICTION ENGINE

Project the **FUTURE** today

Li Zhang, CTO @ Prophet42, Inc.

OUR CUSTOMERS

- MOVIE CRITICS

'runtime', 'Drama', 'Biography', 'R', 'History', 'War', 'PG', 'Comedy', 'Horror' (9)

Luther

Assisted rating before you watch!*

- MOVIE PRODUCERS

'budget', 'Adventure', 'PG', 'runtime', 'R' (5)

Luther

Revenue forecast to plan your next masterpiece!*

***Individual results may vary**

THE MAKING OF LUTHER

- Data acquisition: 10,000 movies from imdb.com from 1985 to 2015

- Data cleaning: df.shape = (3902, 35)

- Determine key features**: df.corr()

** Key features MUST be available ***before*** movie hits theaters
For example, "rating" cannot be used to predict revenue

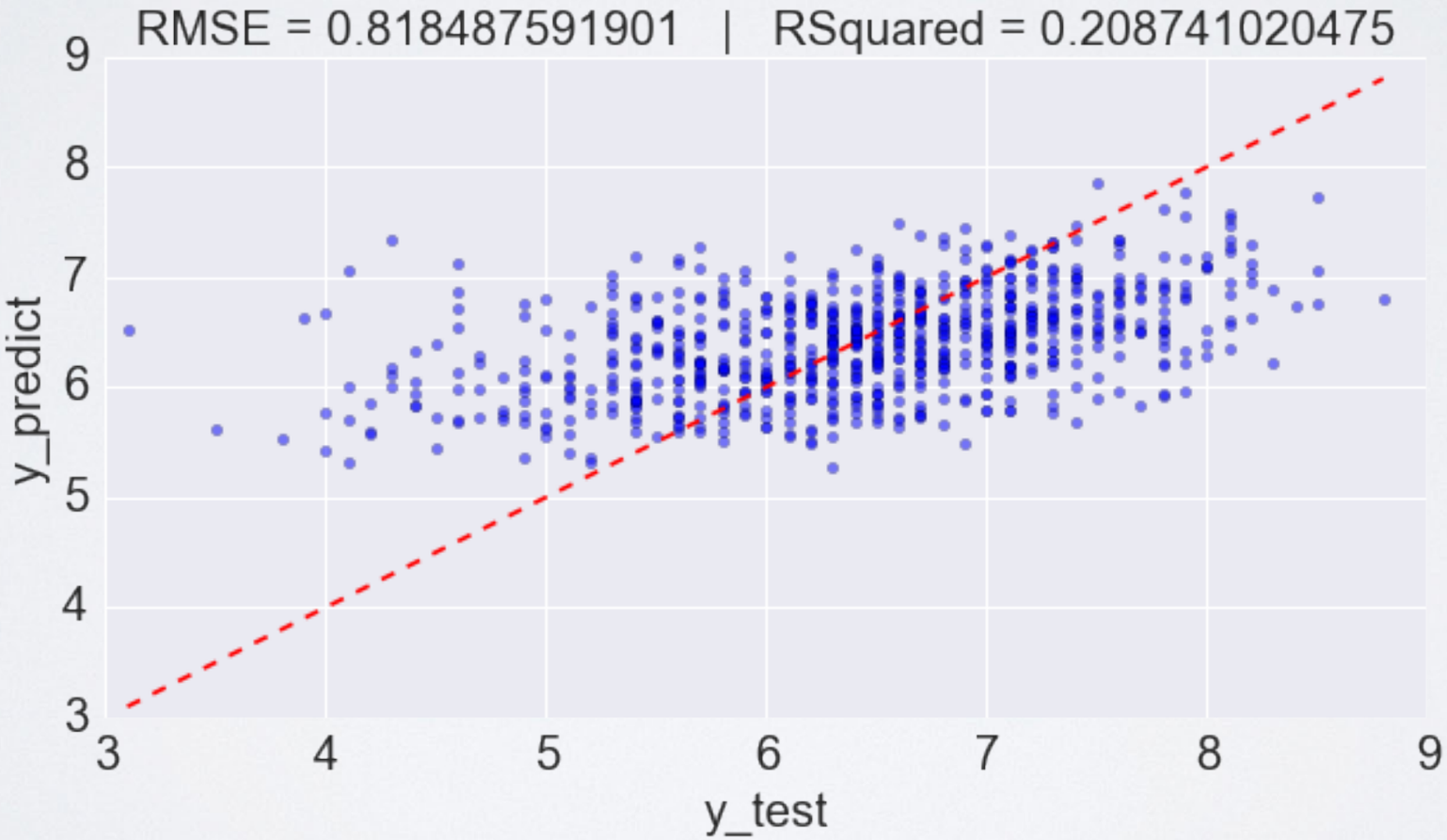
ASSISTED RATING

'runtime', 'Drama', 'Biography', 'R', 'History', 'War', 'PG', 'Comedy', 'Horror' (9)

Luther

Assisted rating before you watch!

step 1. standard linear regression



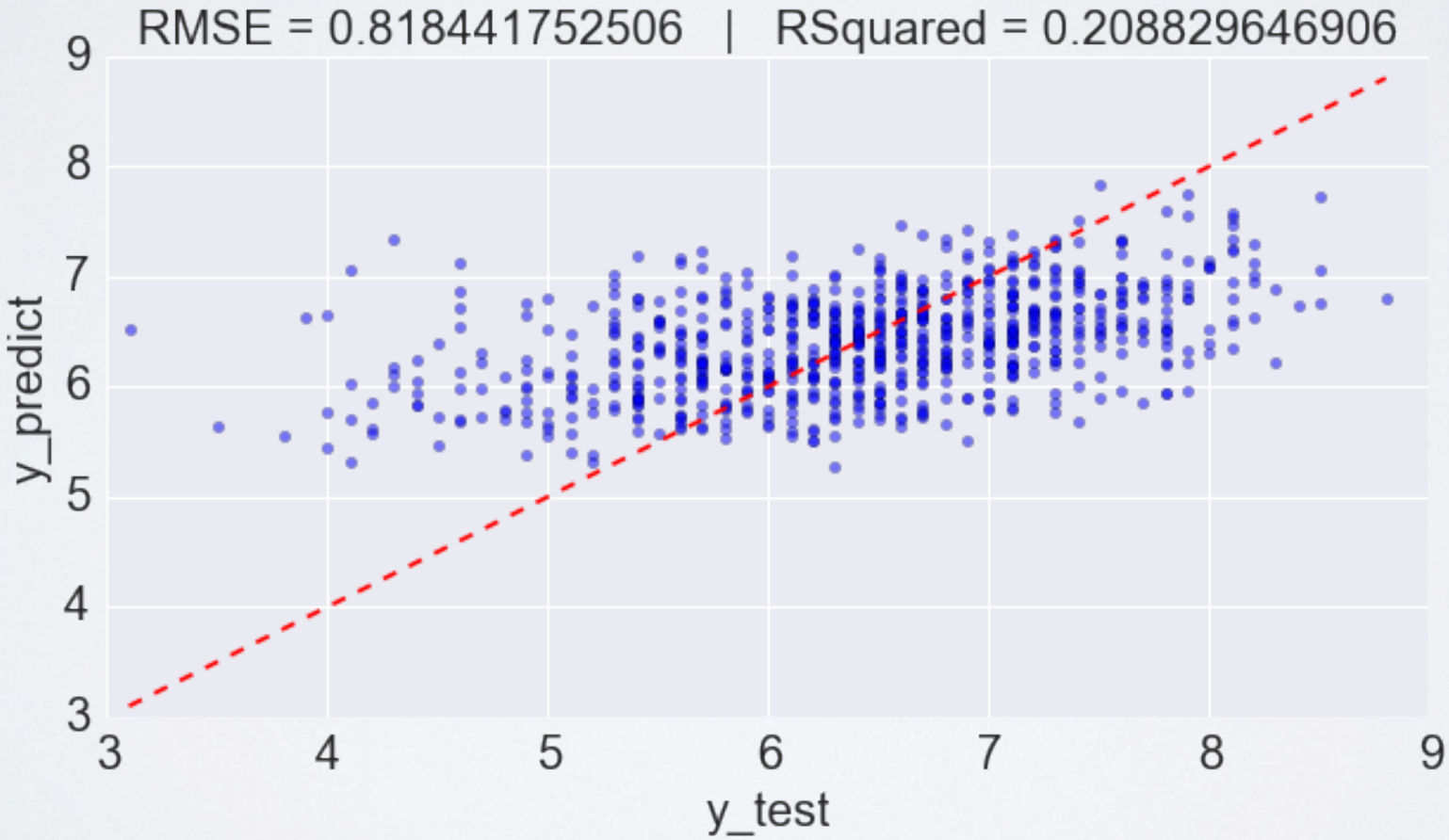
ASSISTED RATING

'runtime', 'Drama', 'Biography', 'R', 'History', 'War', 'PG', 'Comedy', 'Horror' (9)

Luther

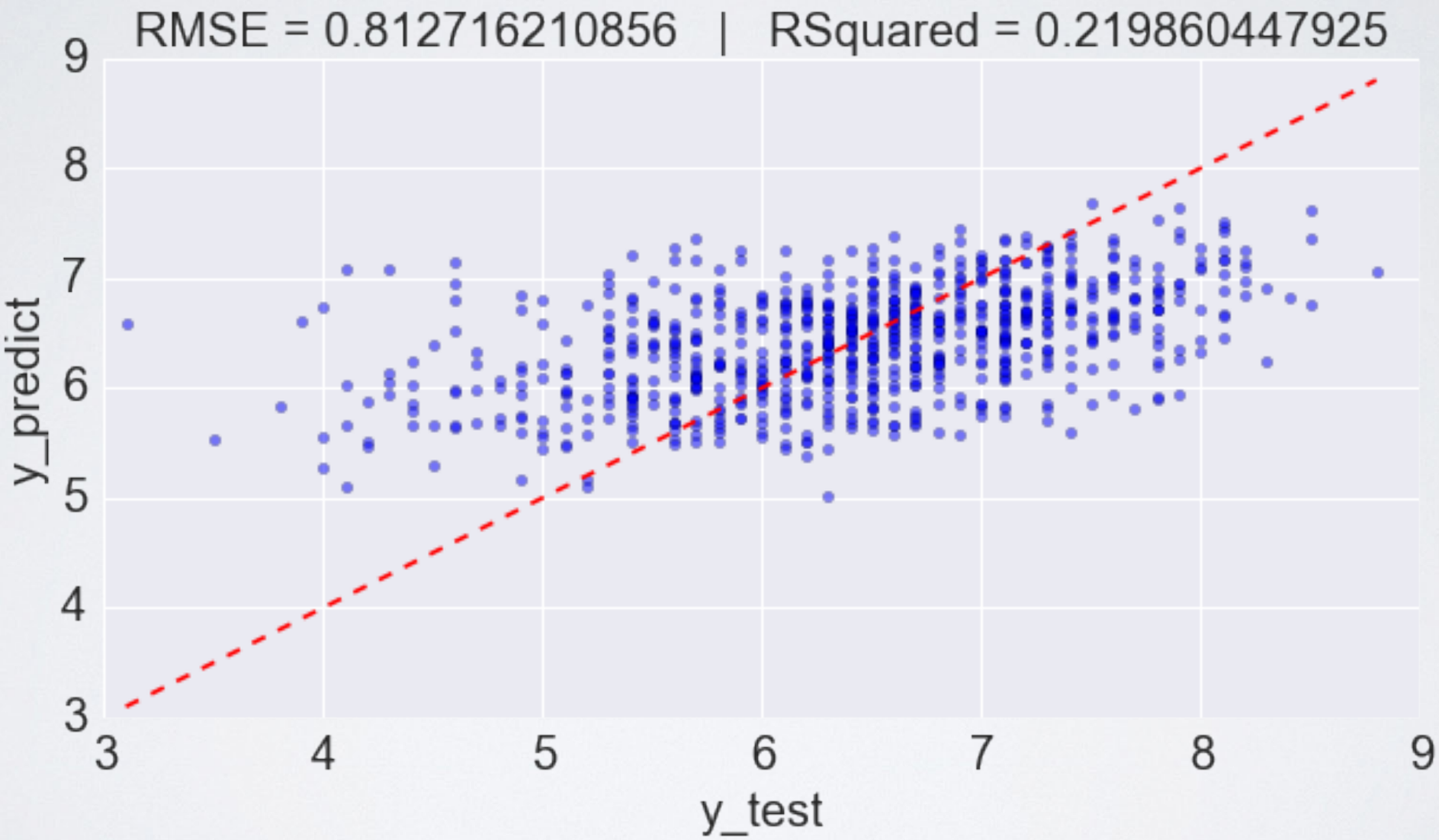
Assisted rating before you watch!

step 2. linear regression with lasso (alphas = [1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 1e2, 1e3, 1e4, 1e5, 1e6, 1e7, 1e8]) —> **best alpha = 0.001**



ASSISTED RATING

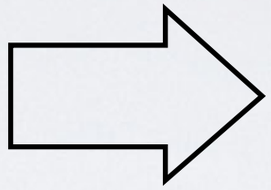
step 3. make_pipeline(PolynomialFeatures(2), Lasso(alpha = .001))



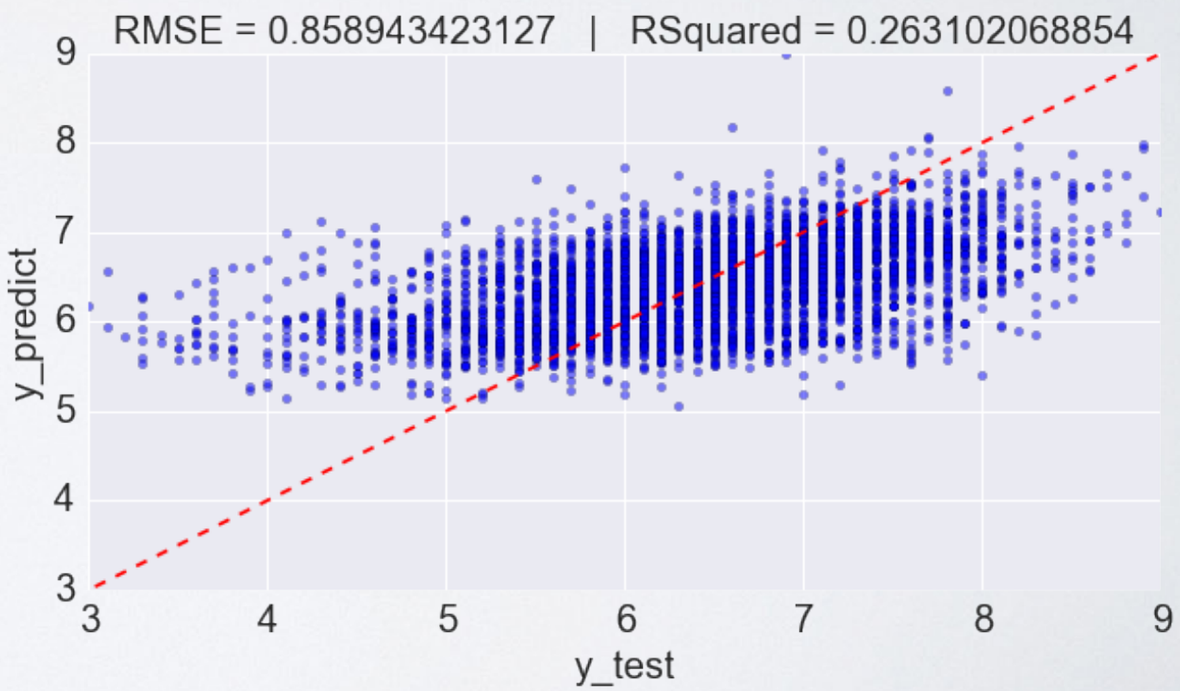
ASSISTED RATING

step 4. Select some of the polynomial features

```
[('1', 0.0),
 ('x0', 0.04658774712573840 5),
 ('x1', 0.0),
 ('x2', 0.0),
 ('x3', 0.0),
 ('x4', 0.0),
 ('x5', 0.0),
('x6', -0.4120398457033980 2),
('x7', 0.2773413915367505 3),
('x8', -0.0),
('x0^2', -6.665739611292522 5e-05),
('x0 x1', -0.0056434742330138061),
('x0 x2', 0.0007392901884530 7918),
('x0 x3', -0.0082820633956669 62),
('x0 x4', -0.0012115946773798392),
('x0 x5', -0.0030767822938704 78),
('x0 x6', -0.0067788572228548701),
('x0 x7', -0.0071362597024425736),
('x0 x8', -0.0027468246290647 57),
```



```
lm1 = smf.ols('y ~ x0 + x6 + x7 + x8 + \
I(x0*x0) + I(x0*x1) + I(x0*x2) + I(x0*x3) + \
I(x0*x4) + I(x0*x5) + I(x0*x6) + I(x0*x7) + \
I(x0*x8) + I(x1**2) + I(x1*x2) + I(x1*x3) + \
I(x1*x6) + I(x1*x7) + I(x1*x8) + I(x2*x6) + \
I(x3*x4) + I(x3*x7) + I(x4*x4) + I(x5*x5) + \
I(x5*x6) + I(x6*x7) + I(x6*x8) + I(x7*x7) + \
I(x7*x8)' , data=df_rating)
```

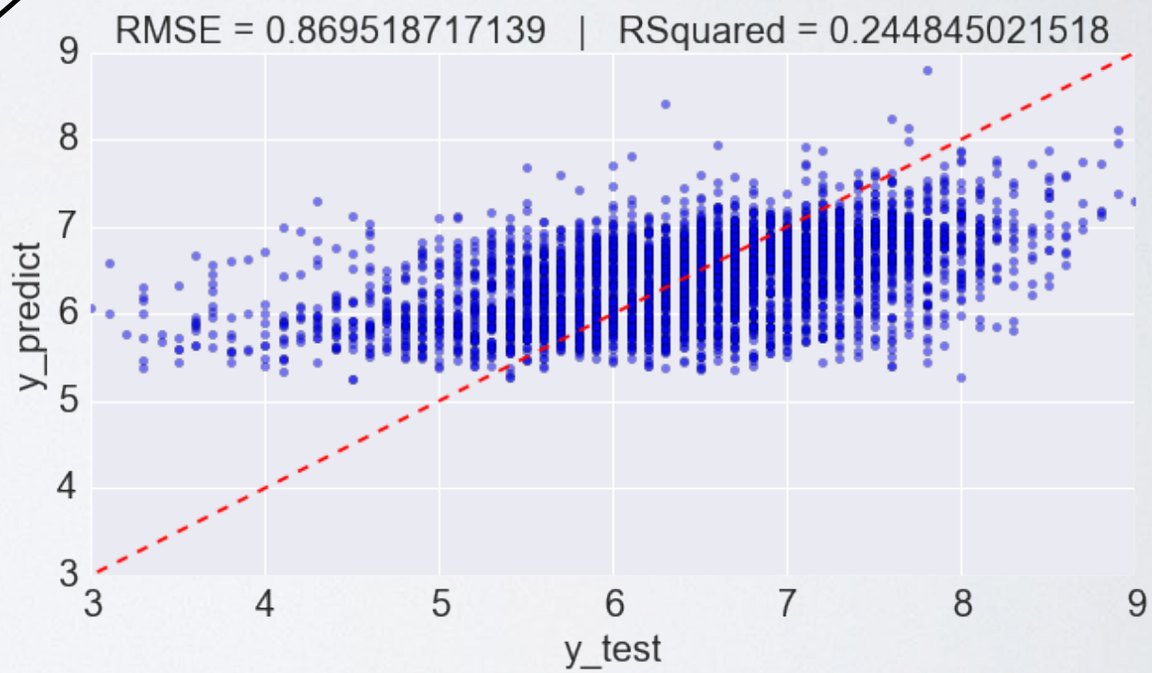


ASSISTED RATING

(*attempted and failed*) step 5. further remove items from the stats model with high p value.

	coef	std err	t	P>|t|	[95.0% Conf. Int.]
Intercept	3.9376	0.376	10.480	0.000	3.201 4.674
x0	0.0292	0.006	4.702	0.000	0.017 0.041
x6	-0.4243	0.208	-2.040	0.041	-0.832 -0.017
x7	0	0	nan	nan	0 0
x8	-0.4746	0.387	-1.225	0.221	-1.234 0.285
I(x0 * x0)	-1.008e-05	2.19e-05	-0.460	0.645	-5.3e-05 3.28e-05
I(x0 * x1)	-0.0068	0.002	-3.354	0.001	-0.011 -0.003
I(x0 * x2)	0.0004	0.003	0.160	0.873	-0.005 0.006
I(x0 * x3)	-0.0054	0.002	-2.580	0.010	-0.009 -0.001
I(x0 * x4)	-0.0032	0.004	-0.821	0.412	-0.011 0.004

```
lm1 = smf.ols('y ~ x0 + x6  + I(x0*x1) + I(x0*x3) + \
                + I(x0*x7)  + I(x1**2) + I(x1*x8)'
                , data=df_rating)
```



ASSISTED RATING

step 6 KFold (n = 5) cross validation

- **0.211041628636**
- **0.238587980349**
- **0.315476940281**
- **0.294975747774**
- **0.233990161157**

ASSISTED RATING

step 1. standard linear regression

step 2. linear regression with lasso and determine alpha

step 3. polynomial feature

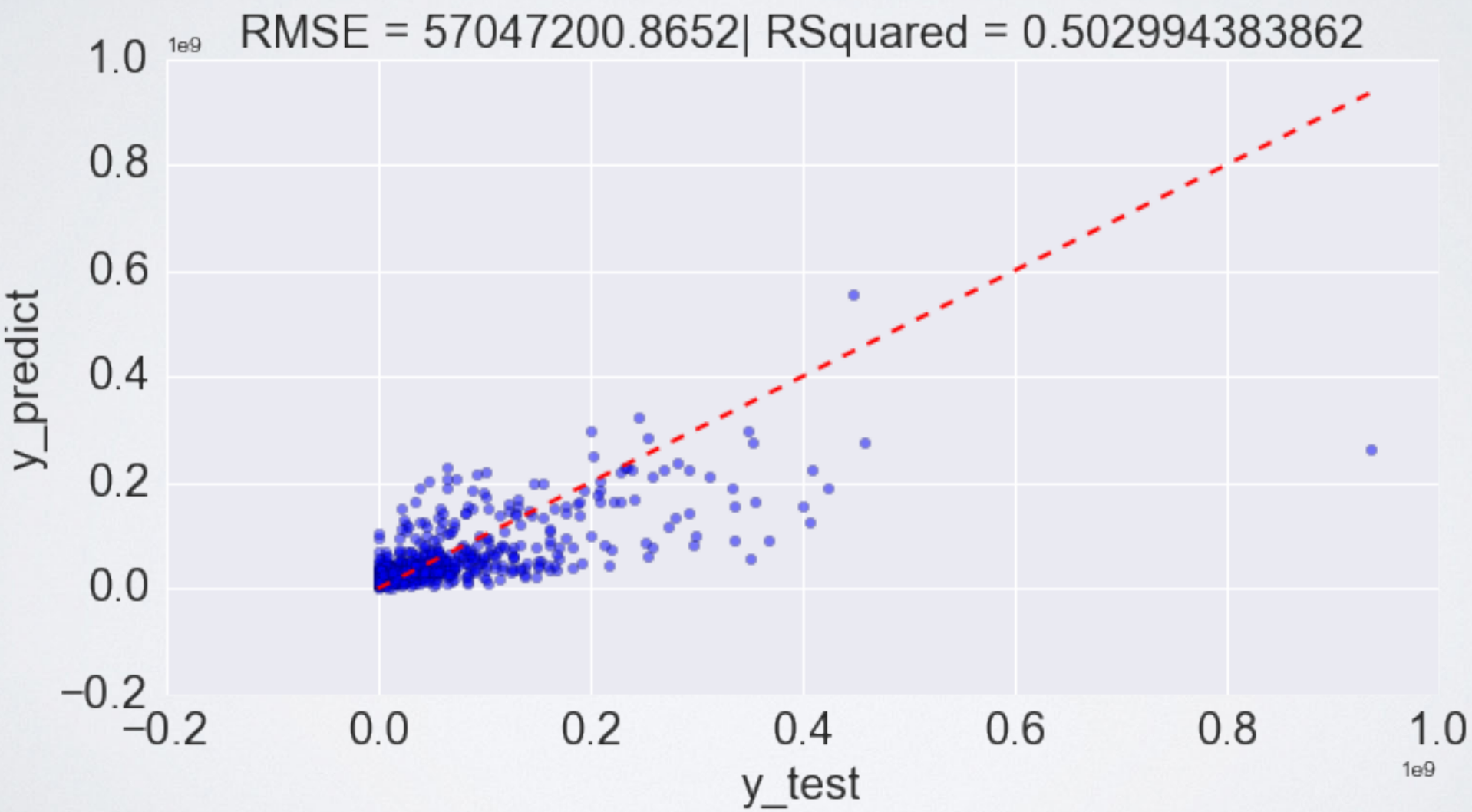
step 4. analyze coefficients from step 3 and create a stats model formula

step 5. simplify model with p-values

step 6. cross validation

REVENUE FORECAST

- **apply the same work flow**
- **optimized at step 3 - polynomial features, step 4 & 5 did NOT improve RSquared**



REVENUE FORECAST

KFold (n = 5) cross validation results

- **0.175148295377**
- **0.44574703386**
- **0.452391592937**
- **0.582333164872**
- **0.513650613406**

LUTHER 2.0 ROADMAP

- **Investigate the anomaly found in KFold cross validation**
- **Verify the residual of the fit is normally distributed**

Thank you

Order today and *live the future*
call 1-800-PROPHET42