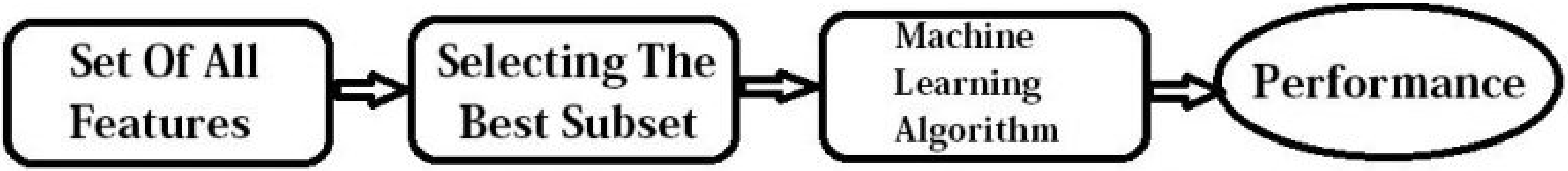
Regression: feature selection

PREPARING FOR MACHINE LEARNING INTERVIEW QUESTIONS IN PYTHON

Lisa Stuart
Data Scientist

Selecting the correct features:

- Reduces overfitting
- Improves accuracy
- Increases interpretability
- Reduces training time



[1] https://www.analyticsindiamag.com/what [2] are [3] feature [4] selection [5] techniques [6] in [7] machine [8] learning/

Feature selection methods

- Filter: Rank features based on statistical performance
- Wrapper: Use an ML method to evaluate performance
- Embedded: Iterative model training to extract features
- Feature importance: tree-based ML models

Compare and contrast methods

Method	Use an ML model	Select best subset	Can overfit
Filter	No	No	No
Wrapper	Yes	Yes	Sometimes
Embedded	Yes	Yes	Yes
Feature importance	Yes	Yes	Yes

Correlation coefficient statistical tests

Feature/Response	Continuous	Categorical
Continuous	Pearson's Correlation	LDA
Categorical	ANOVA	Chi-Square

Filter functions

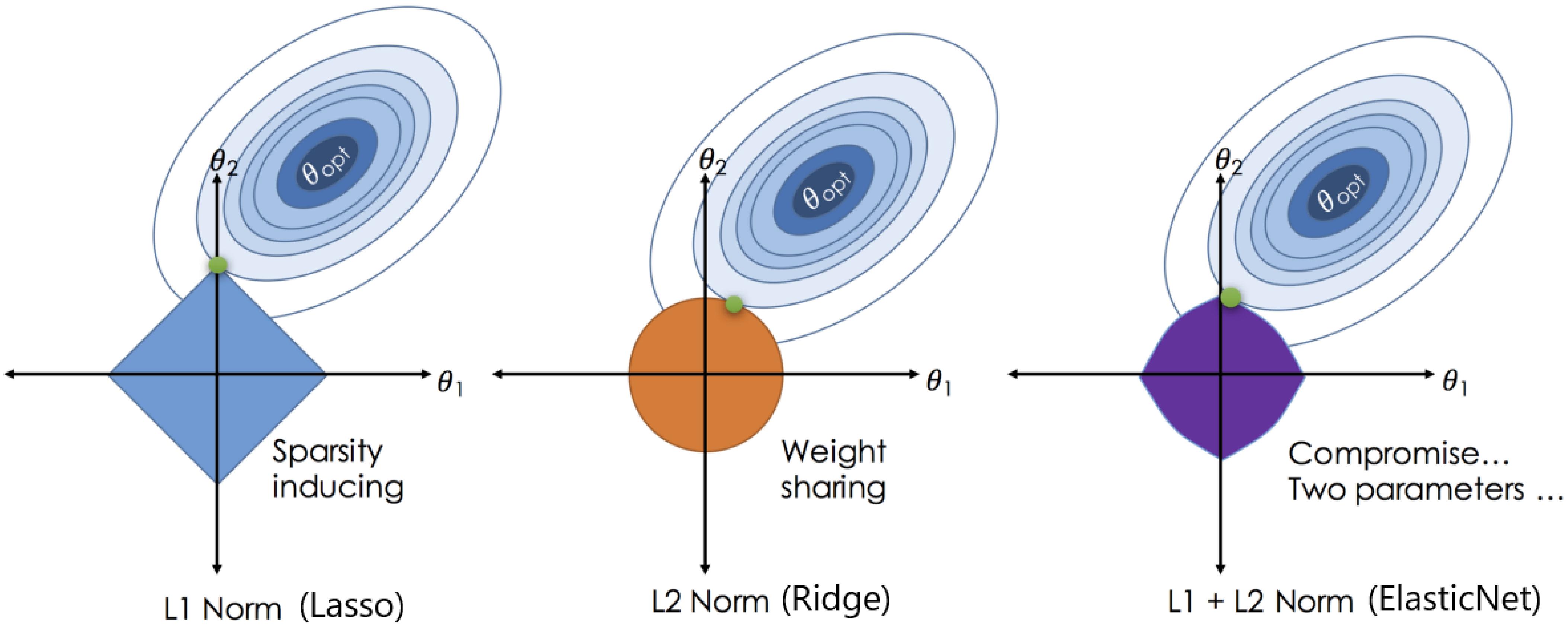
Function	returns
`df.corr()`	Pearson's correlation matrix
`sns.heatmap(corr_object)`	heatmap plot
`abs()`	absolute value

Wrapper methods

1. Forward selection (LARS-least angle regression)
   - Starts with no features, adds one at a time
2. Backward elimination
   - Starts with all features, eliminates one at a time
3. Forward selection/backward elimination combination (bidirectional elimination)
4. Recursive feature elimination
   - RFECV

Embedded methods

1. Lasso Regression
2. Ridge Regression
3. ElasticNet



Tree-based feature importance methods

- Random Forest --> `sklearn.ensemble.RandomForestRegressor`
- Extra Trees --> `sklearn.ensemble.ExtraTreesRegressor`
- After model fit --> `tree_mod.feature_importances_`

Function	returns
`sklearn.svm.SVR`	support vector regression estimator
`sklearn.feature_selection.RFECV`	recursive feature elimination with cross-val
`rfe_mod.support_`	boolean array of selected features
`ref_mod.ranking_`	feature ranking, selected=1
`sklearn.linear_model.LinearRegression`	linear model estimator
`sklearn.linear_model.LarsCV`	least angle regression with cross-val
`LarsCV.score`	r-squared score
`LarsCV.alpha_`	estimated regularization parameter

Let's practice!

PREPARING FOR MACHINE LEARNING INTERVIEW QUESTIONS IN PYTHON

Regression: regularization

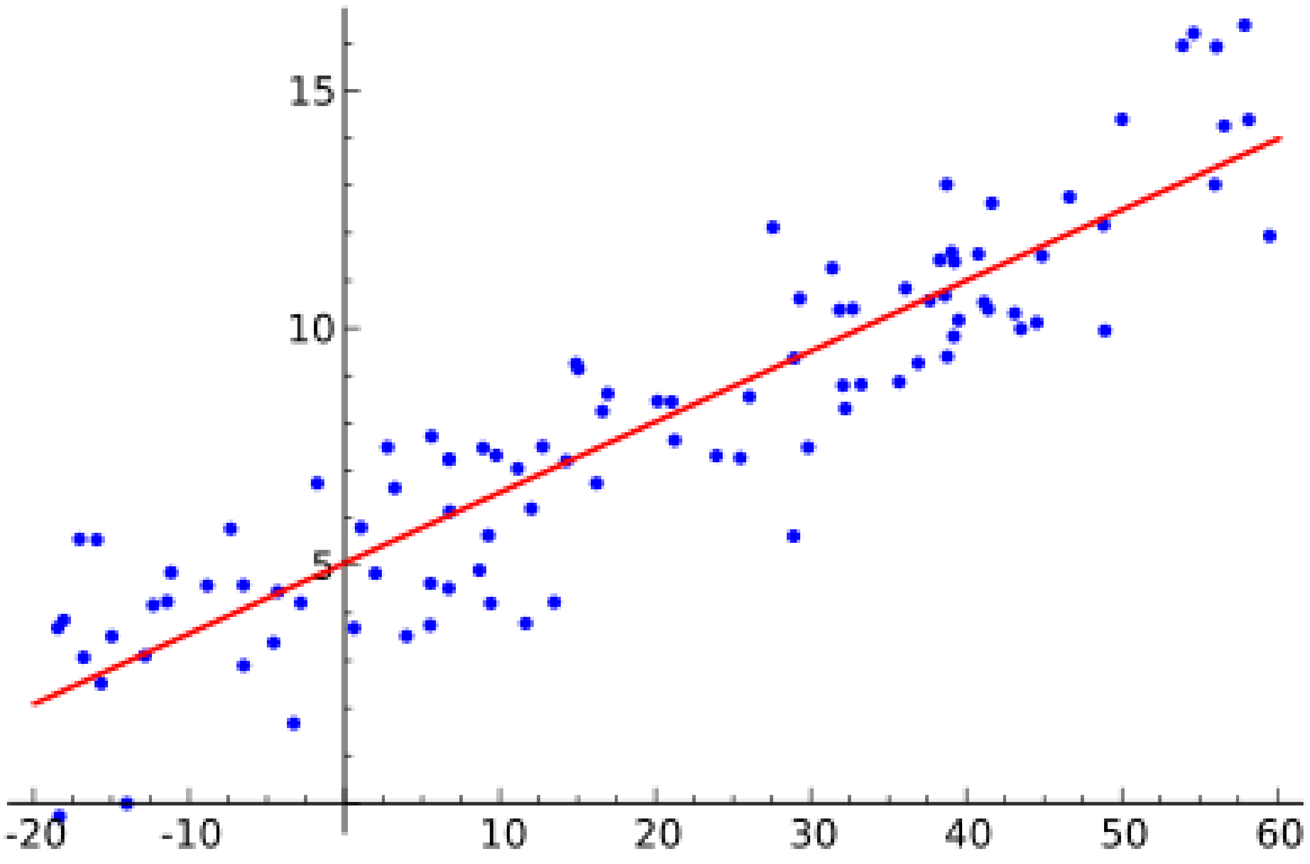
PREPARING FOR MACHINE LEARNING INTERVIEW QUESTIONS IN PYTHON

Lisa Stuart
Data Scientist

Regularization algorithms

- Ridge regression
- Lasso regression
- ElasticNet regression

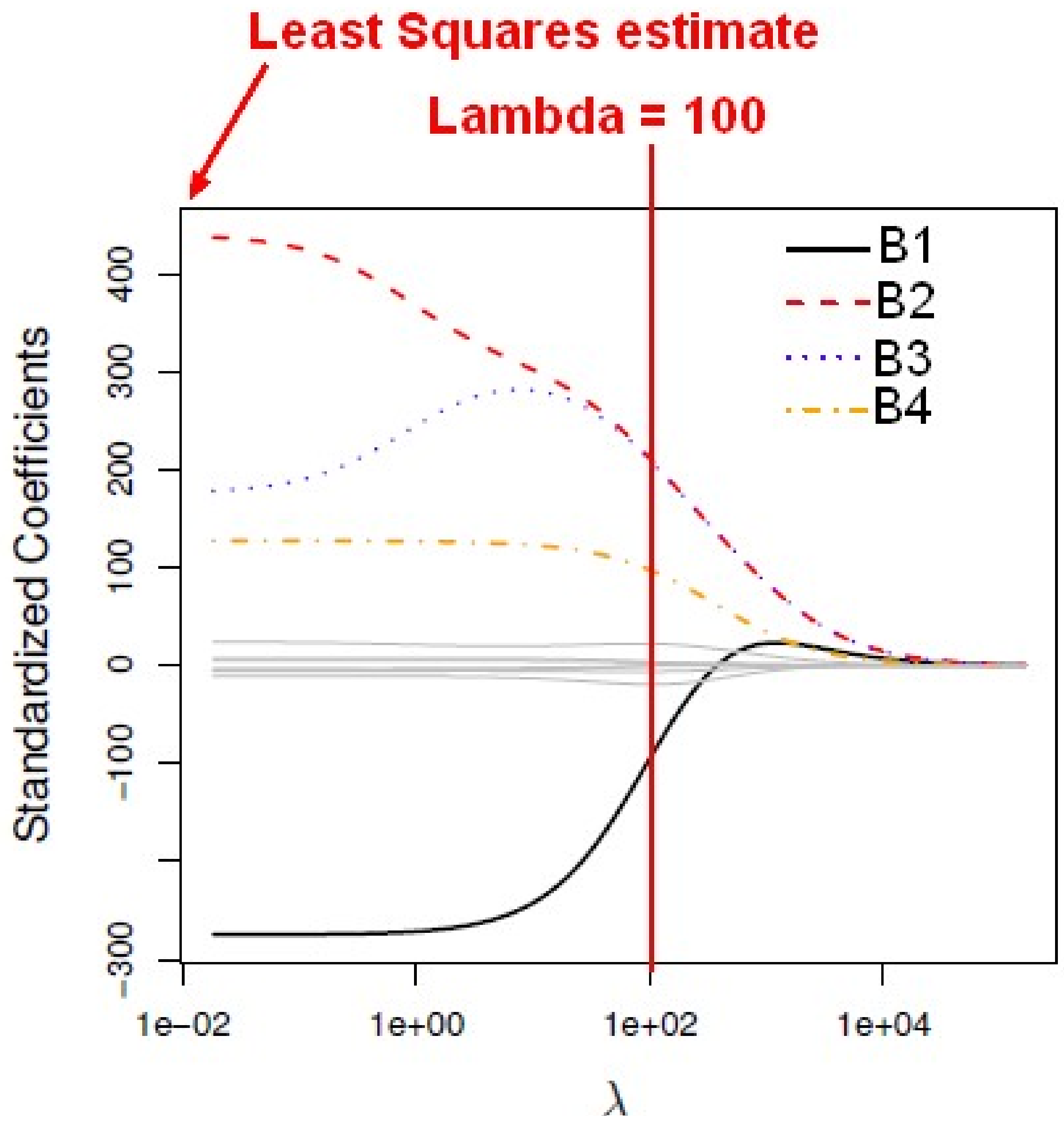
Ordinary least squares



$$\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

[1] https://en.wikipedia.org/wiki/Linear_regression#Simple_and_multiple_linear_regression

Ridge loss function

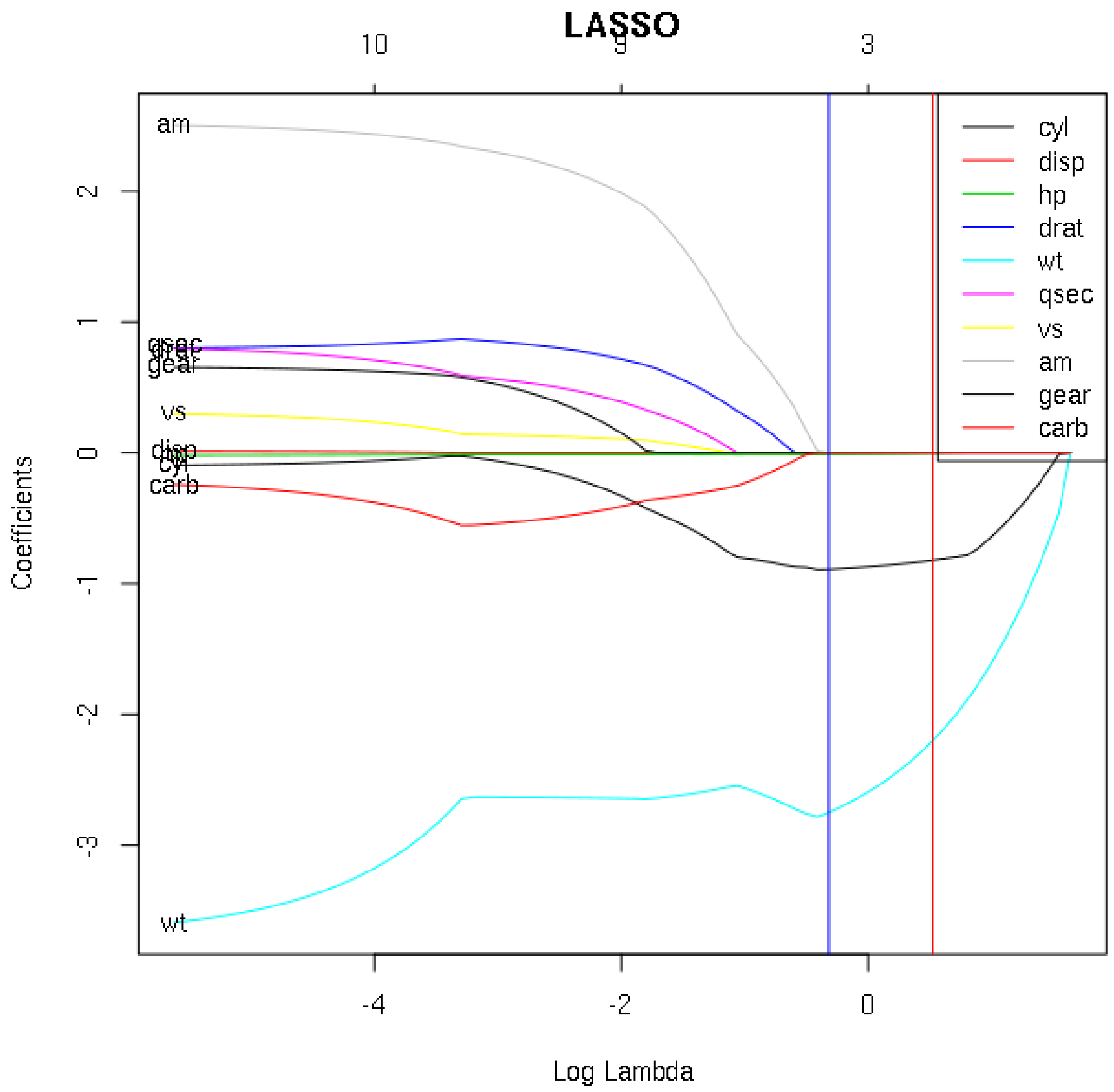


Ridge penalty term

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \boxed{\lambda \sum_{j=1}^{p} \beta_j^2}$$

[1] https://gerardnico.com/data_mining/ridge_regression#tuning_parameter_math_lambdamath

Lasso loss function



Lasso penalty term

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \boxed{\lambda \sum_{j=1}^{p}|\beta_j|}$$

[1] https://stats.stackexchange.com/questions/155192/why [2] discrepancy [3] between [4] lasso [5] and [6] randomforest

Ridge vs lasso

Regularization	L1 (Lasso)	L2 (Ridge)
penalizes	sum of absolute value of coefficients	sum of squares of coefficients
solutions	sparse	non-sparse
number of solutions	multiple	one
feature selection	yes	no
robust to outliers?	yes	no
complex patterns?	no	yes

ElasticNet

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \underbrace{\boxed{\lambda\left((1-\alpha)\sum_{j=1}^{p}|\beta_j| + \alpha\sum_{j=1}^{p}\beta_j^2\right)}}_{}$$

ElasticNet penalty term

Regularization with Boston housing data

Features	CHAS	NOX	RM
Coefficient estimates	2.7	-17.8	3.8
Regularized coefficient estimates	0	0	0.95

Regularization functions

```
# Lasso estimator
sklearn.linear_model.Lasso

# Lasso estimator with cross-validation
sklearn.linear_model.LassoCV

# Ridge estimator
sklearn.linear_model.Ridge

# Ridge estimator with cross-validation
sklearn.linear_model.RidgeCV

# ElasticNet estimator
sklearn.linear_model.ElasticNet
```

```
# ElasticNet estimator with cross-validation
sklearn.linear_model.ElasticNetCV

# Train/test split
sklearn.model_selection.train_test_split

# Mean squared error
sklearn.metrics.mean_squared_error(y_test,
                                   predict(X_test))
# Best regularization parameter
mod_cv.alpha_

# Array of log values
alphas=np.logspace(-6, 6, 13)
```

Let's practice!

PREPARING FOR MACHINE LEARNING INTERVIEW QUESTIONS IN PYTHON

Classification: feature engineering

PREPARING FOR MACHINE LEARNING INTERVIEW QUESTIONS IN PYTHON

Lisa Stuart
Data Scientist

Feature engineering...why?

- Extracts additional information from the data
- Creates additional relevant features
- One of the most effective ways to improve predictive models

Benefits of feature engineering

- Increased predictive power of the learning algorithm
- Makes your machine learning models perform even better!

Types of feature engineering

- Indicator variables
- Interaction features
- Feature representation

Indicator variables

- Threshold indicator
  - age: high school vs college
- Multiple features
  - used as a flag
- Special events
  - black Friday
  - Christmas
- Groups of classes
  - website traffic paid flag
    - Google adwords{4}}
    - Facebook ads

Interaction features

- Sum
- Difference
- Product
- Quotient
- Other mathematical combos

Feature representation

- Datetime stamps
  - Day of week
  - Hour of day
- Grouping categorical levels into 'Other'
- Transform categorical to dummy variables
  - (k - 1) binary columns

Different categorical levels

- Training data:
  - model trained with [red, blue, green]
- Test data:
  - model test with [red, green, yellow]
  - additional color not seen in training
  - one color missing
- Robust one-hot encoding

[1] https://blog.cambridgespark.com/robust [2] one [3] hot [4] encoding [5] in [6] python [7] 3e29bfcec77e

Debt to income ratio

- `Monthly Debt`
- `Annual Income/12`

Feature engineering functions

Function	returns
`sklearn.linear_model.LogisticRegression`	logistic regression
`sklearn.model_selection.train_test_split`	train/test split function
`sns.countplot(x='Loan Status', data=data)`	bar plot
`df.drop(['Feature 1', 'Feature 2'], axis=1)`	drops list of features
`df["Loan Status"].replace({'Paid': 0, 'Not Paid': 1})`	`Loan Status` as integers
`pd.get_dummies()`	k - 1 binary features
`sklearn.metrics.accuracy_score(y_test, predict(X_test))`	model accuracy

An excellent tutorial:

Handling Categorical Data in Python

Learn the common tricks to handle categorical data and preprocess it to build machine learning models!

If you are familiar with machine learning, you will probably have encountered categorical features in many datasets. These generally include different categories or levels associated with the observation, which are non-numerical and thus need to be converted so the computer can process them.

Datacamp article: categorical data

Let's practice!

PREPARING FOR MACHINE LEARNING INTERVIEW QUESTIONS IN PYTHON

Ensemble methods

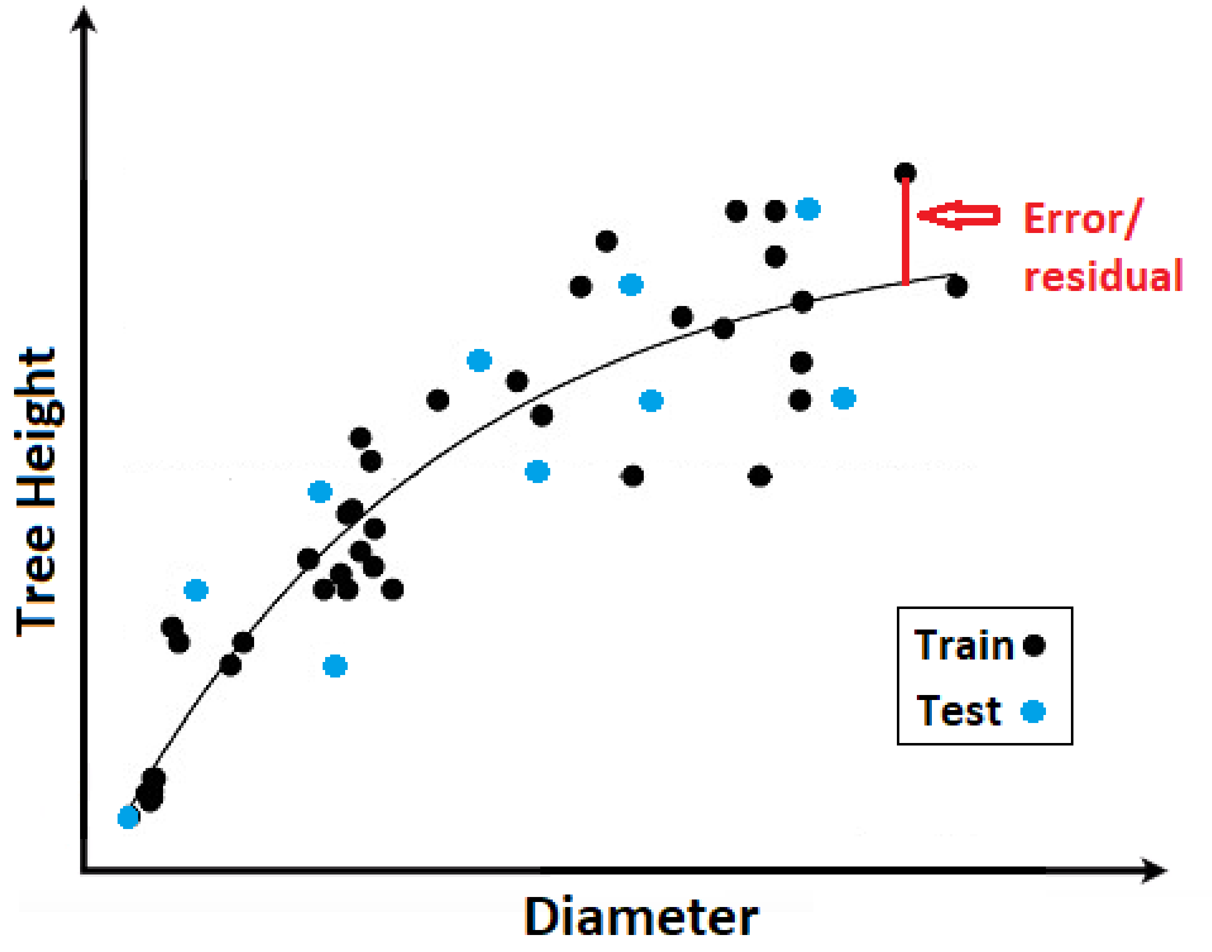
PREPARING FOR MACHINE LEARNING INTERVIEW QUESTIONS IN PYTHON

Lisa Stuart
Data Scientist

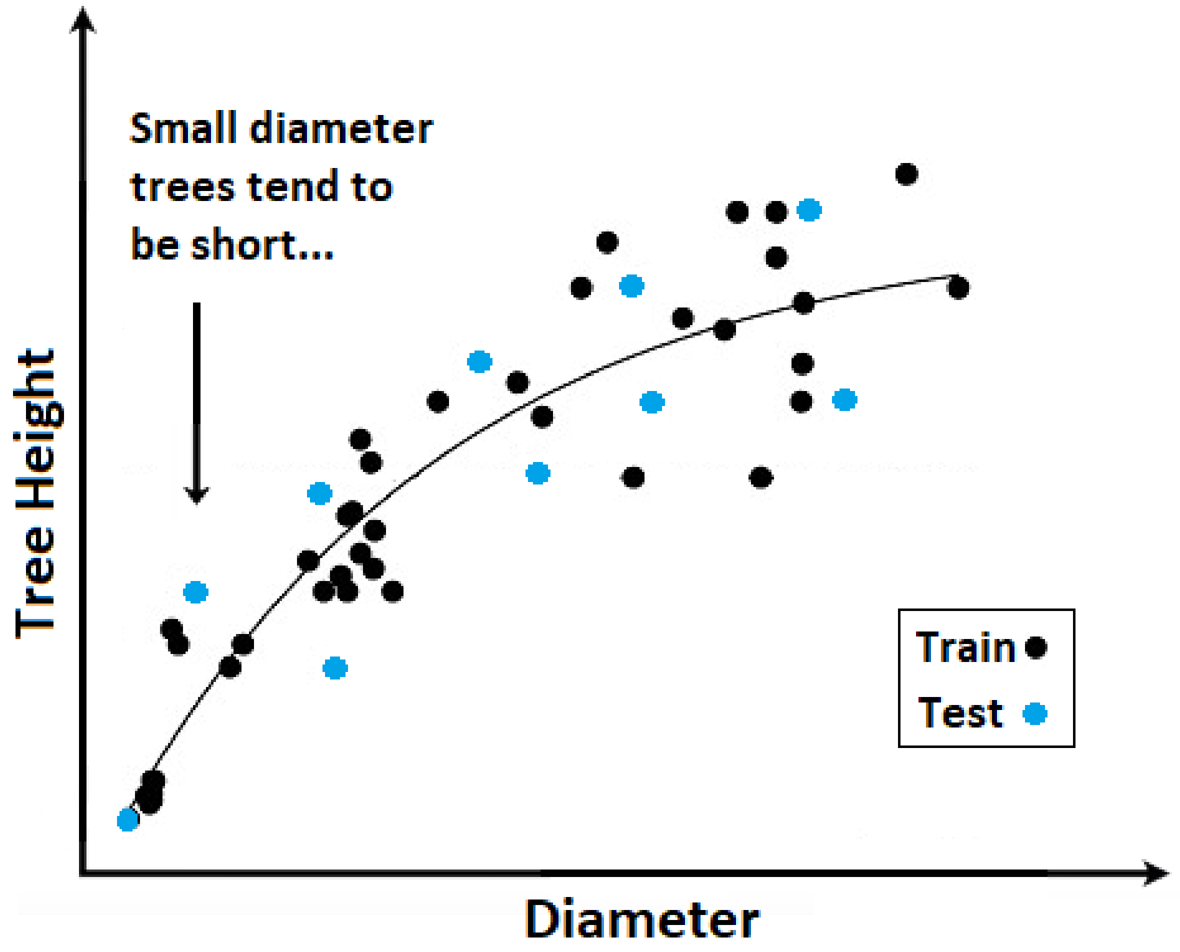
Ensemble learning techniques

- **B**ootstrap **Agg**regation
- Boosting
- Model stacking

Error measurement

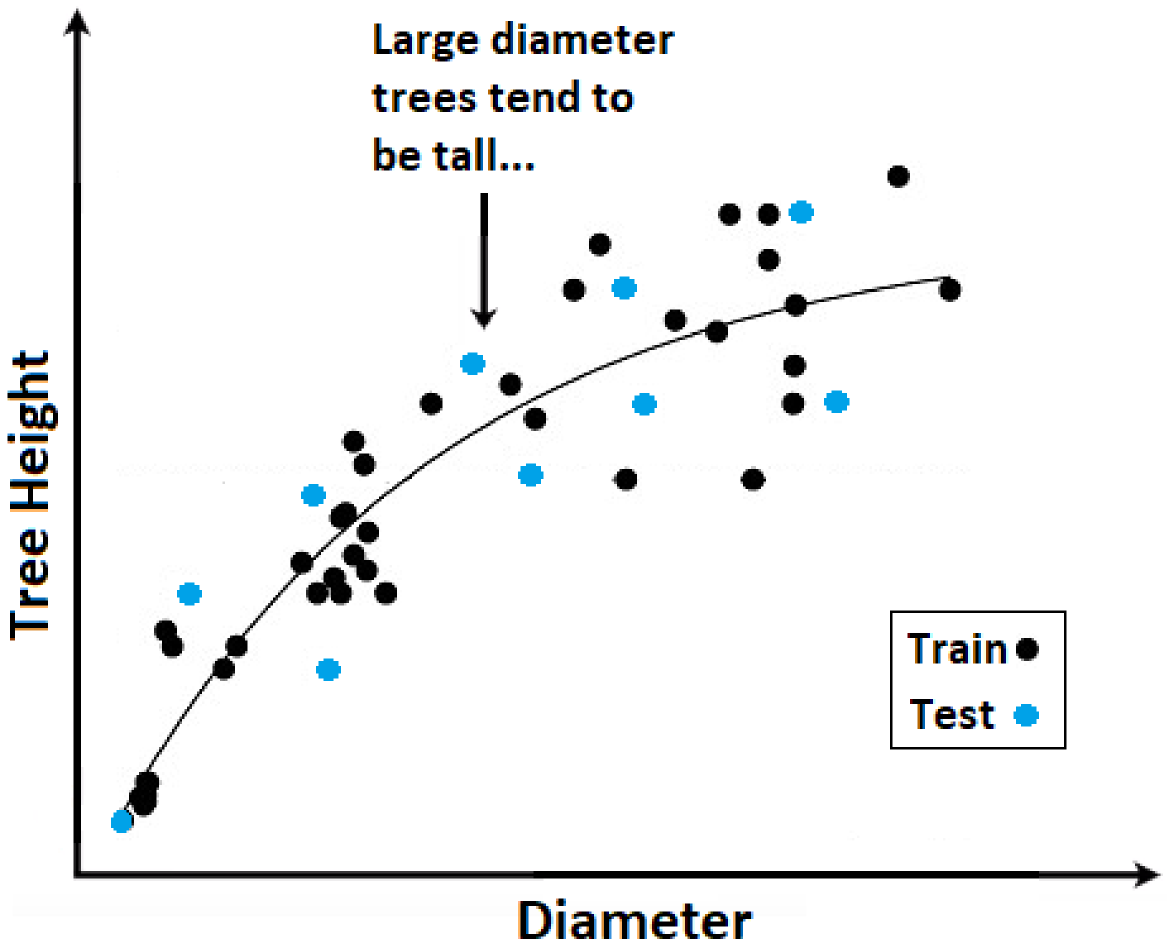


Short trees

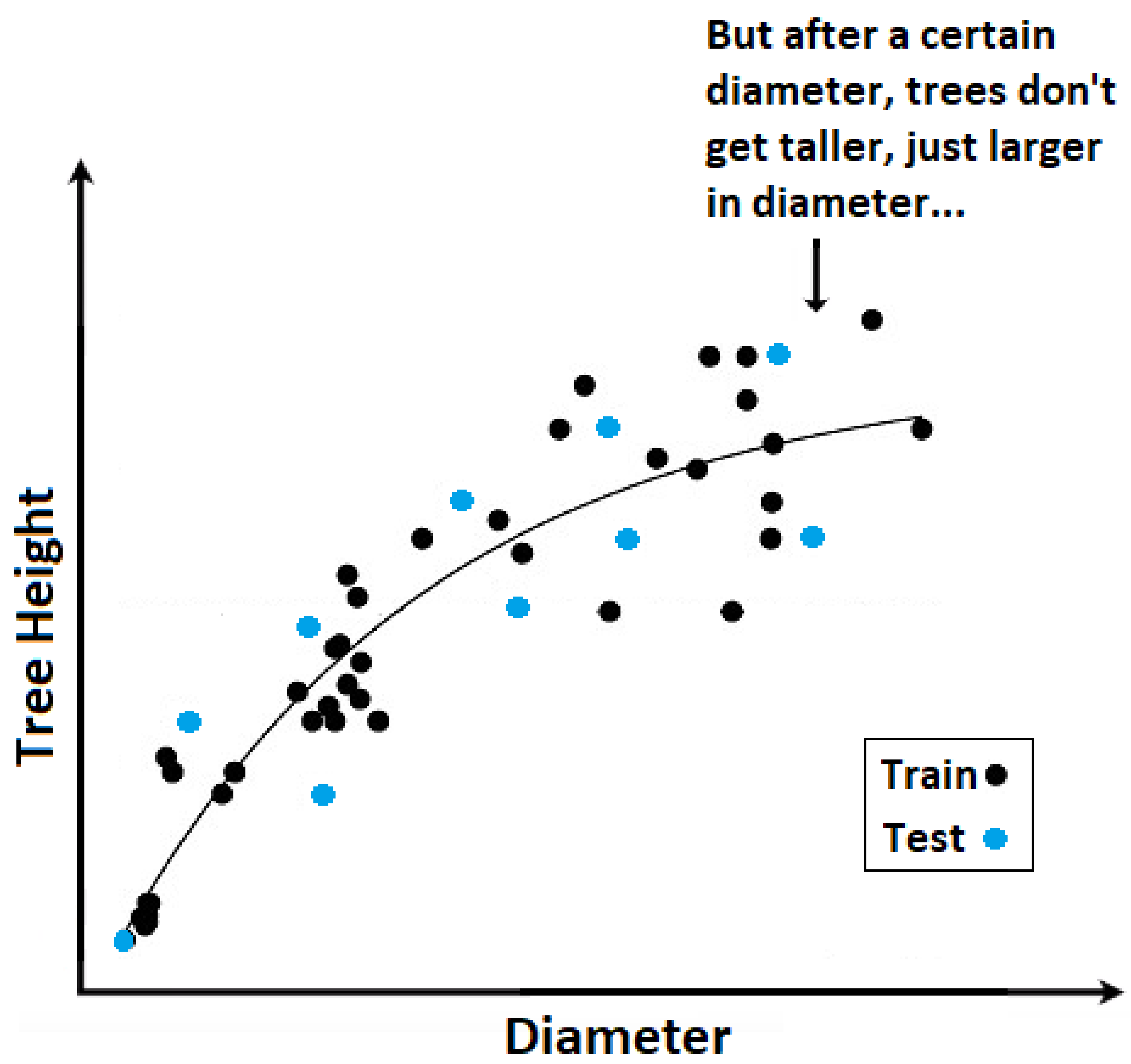


Small diameter
trees tend to
be short...
Tree Height
Diameter
Train
Test

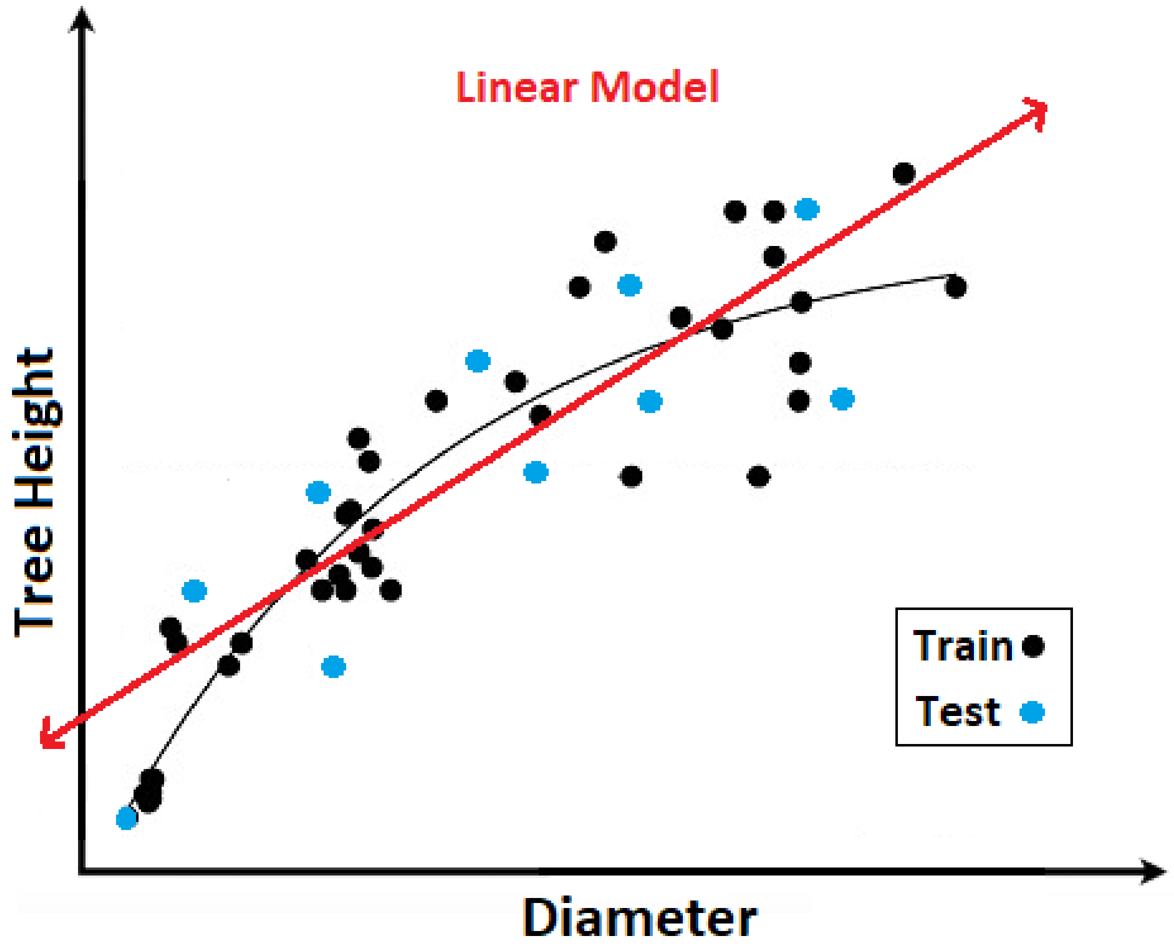
Tall trees


Large diameter
trees tend to
be tall...
Tree Height
Diameter
Train
Test

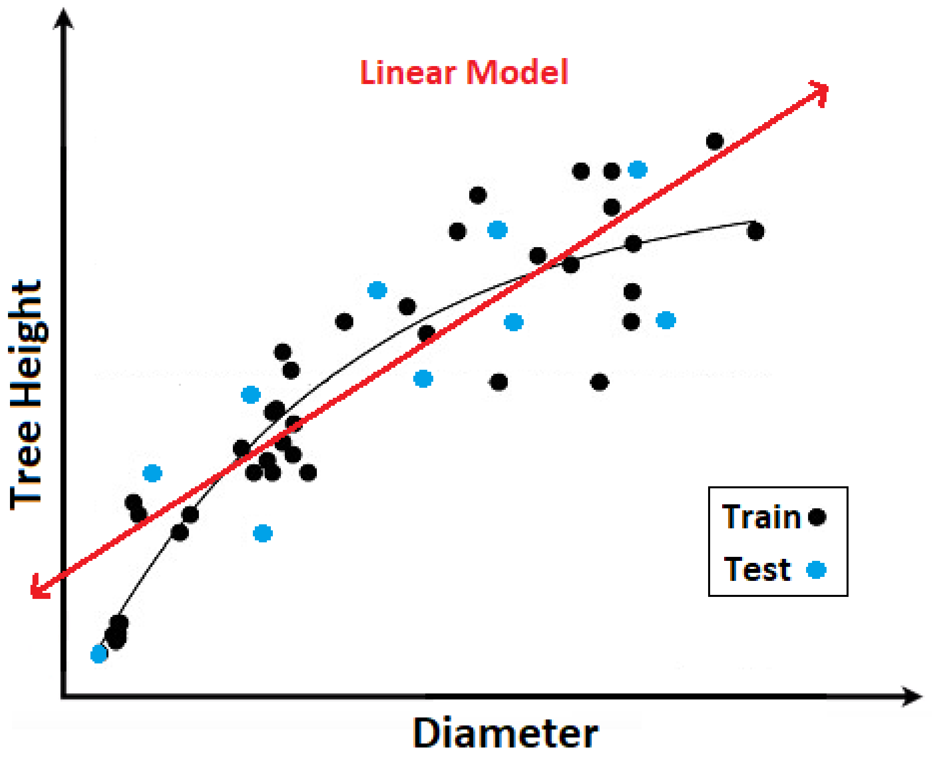
Fat trees


But after a certain diameter, trees don't get taller, just larger in diameter...
Tree Height
Diameter
Train
Test

Linear model



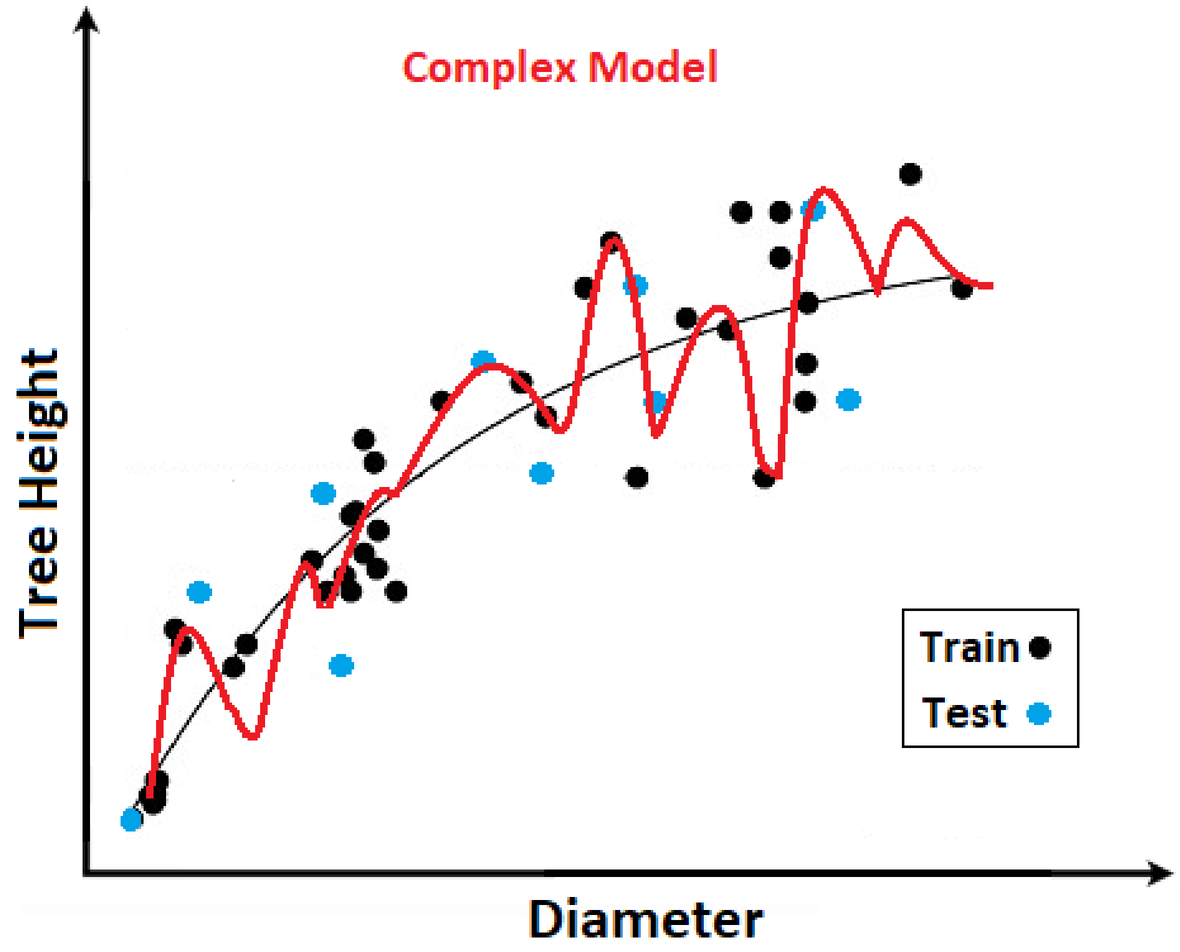
Bias



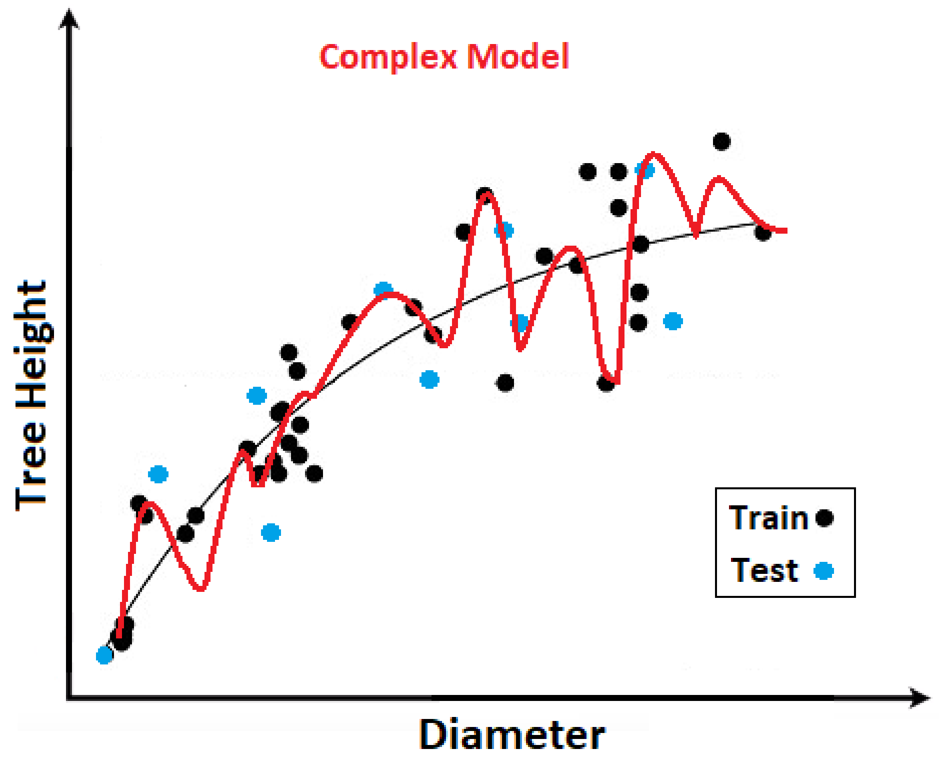
Linear relationship assumption (incorrect)

- High bias
- Underfitting
- Poor model generalization
- Increasing complexity decreases bias

Complex model



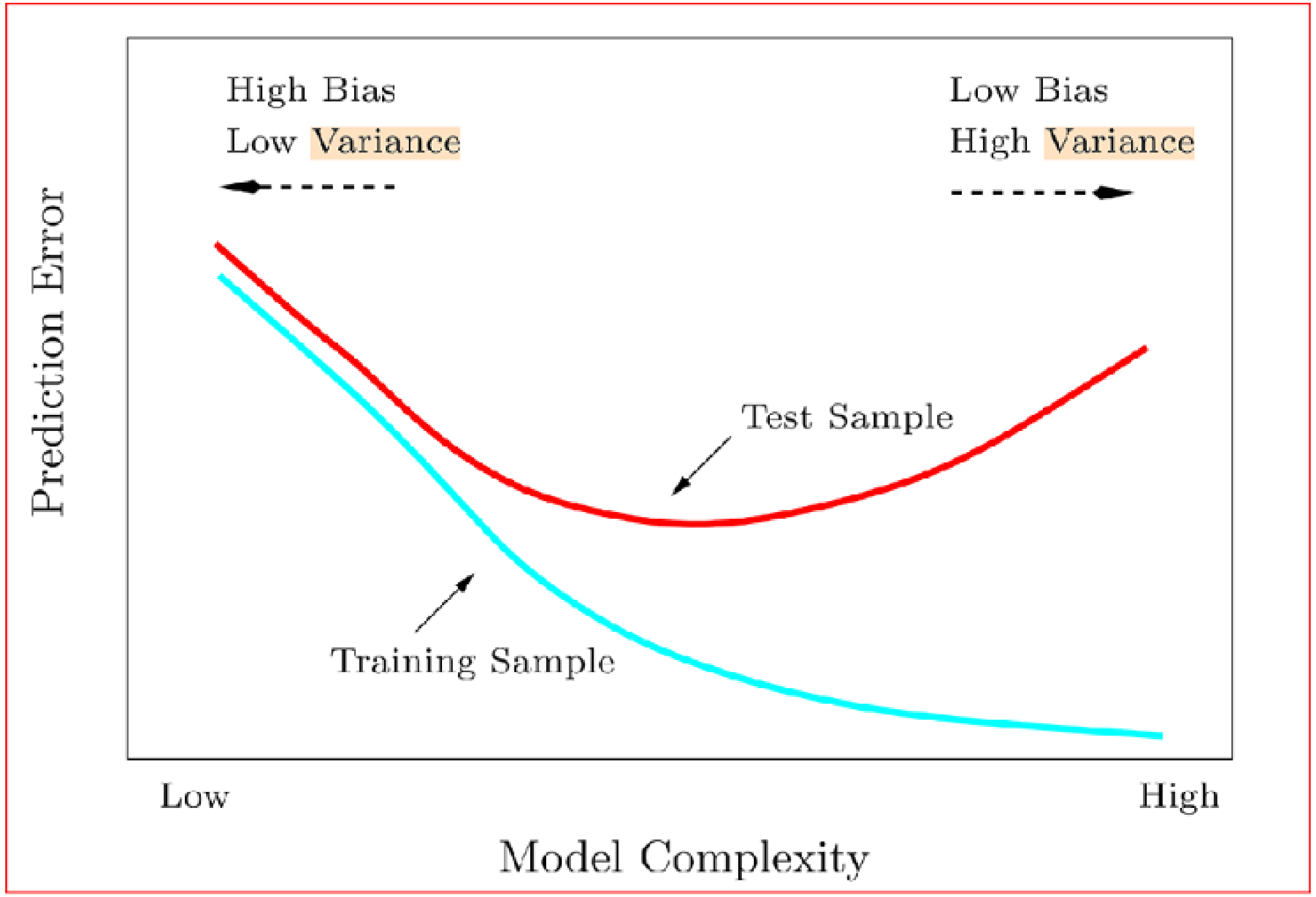
Variance



High complexity models:

- High variance
- Overfitting
- Poor model generalization

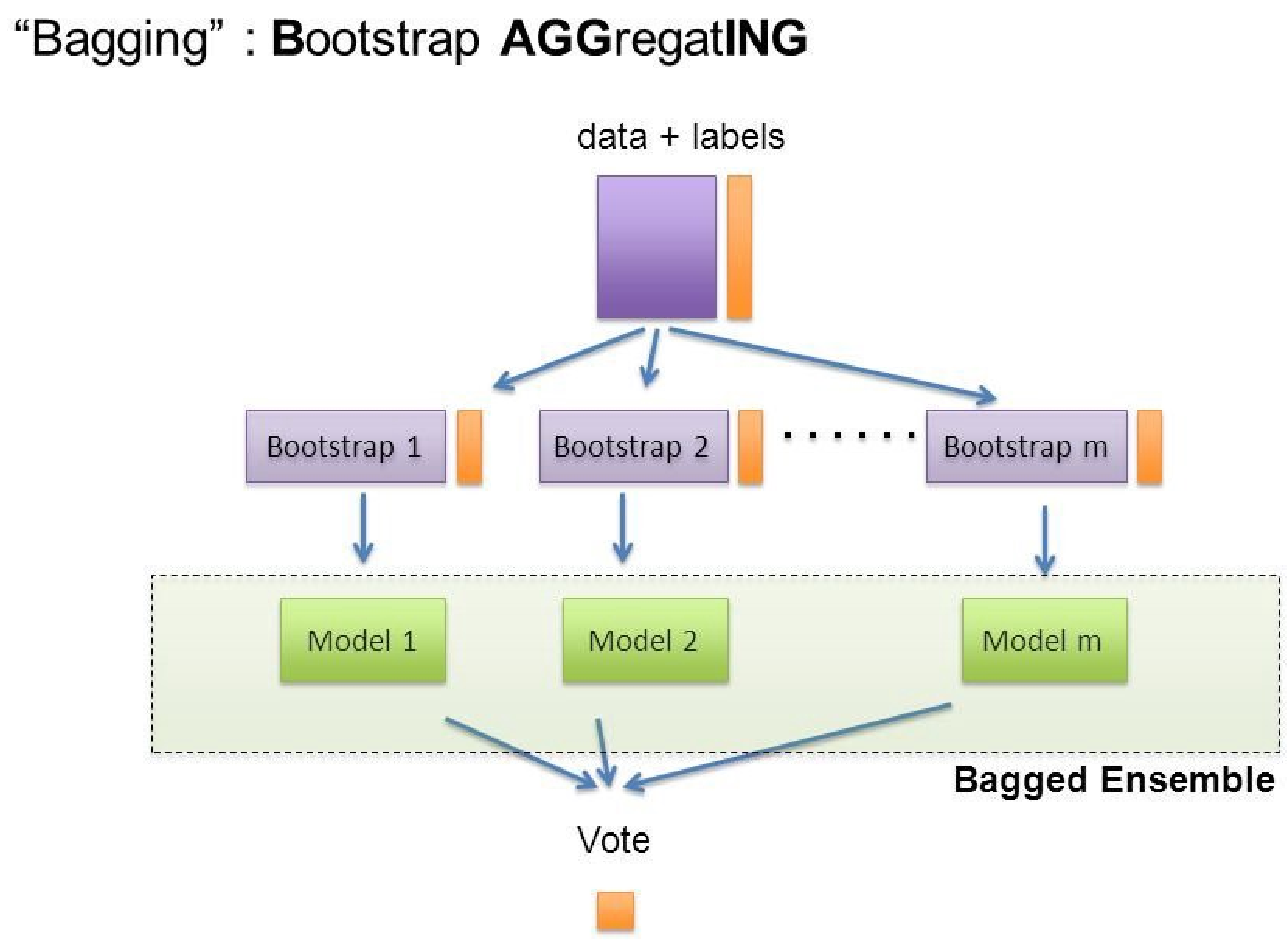
Bias-Variance Trade-Off



[1] Source: Elements of Statistical Learning by Trevor Hastie, Robert Tibshirani and Jerome Friedman

Bagging (Bootstrap aggregation)

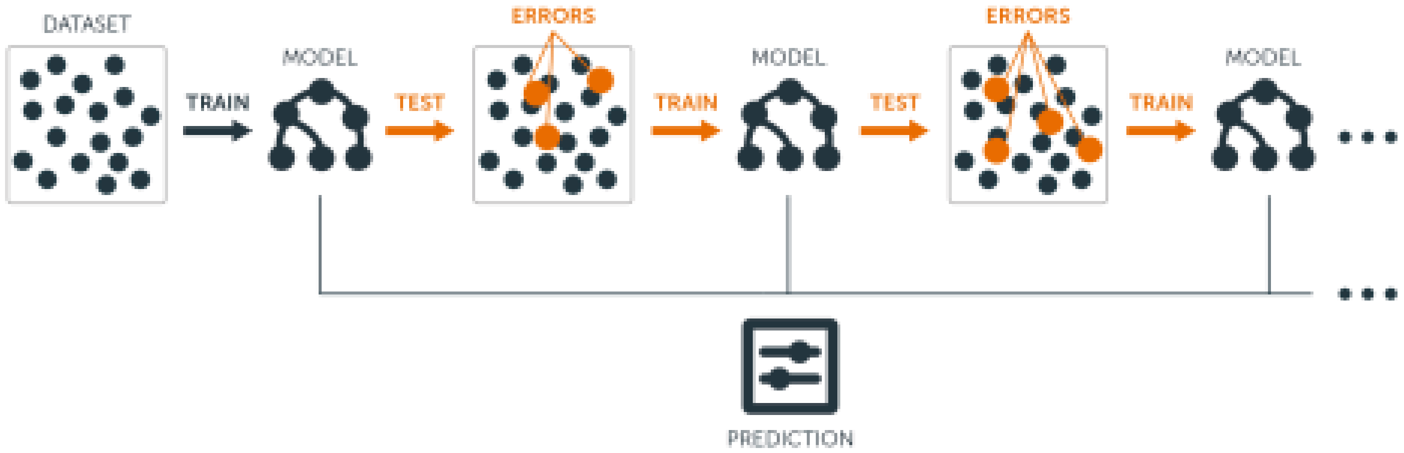
- Bootstrapped samples
  - Subset selected with replacement
  - Same row of data may be chosen
- Model built for each sample
- Average the output
- Reduces variance



[1] https://medium.com/@rrfd/boosting [2] bagging [3] and [4] stacking [5] ensemble [6] methods [7] with [8] sklearn [9] and [10] mlens [11] a455c0c982de

Boosting

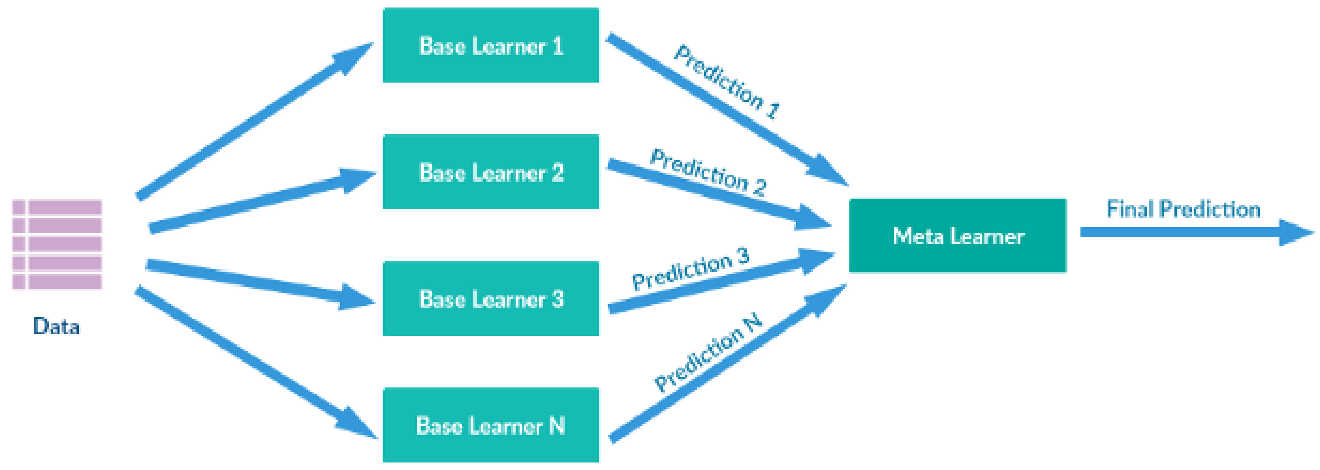
- Multiple models built sequentially
- Incorrect predictions are weighted
- Reduces bias



[1] https://blog.bigml.com/2017/03/14/introduction [2] to [3] boosted [4] trees/

Model stacking

- Model 1 predictions
- Model 2 predictions...
- Model N predictions
- Stack for highest accuracy model
  - Uses base model (Model N) predictions as input to 2nd level model



[1] http://supunsetunga.blogspot.com/

Vecstack package

```
# import modules
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import AdaBoostClassifier
from xgboost import XGBClassifier
from vecstack import stacking

# Create list: stacked_models
stacked_models = [BaggingClassifier(n_estimators=25, random_state=123), AdaBoostClassifier(n_estimators=25, random_state=123)]

# Stack the models: stack_train, stack_test
stack_train, stack_test = stacking(stacked_models, X_train, y_train, X_test, regression=False, mode='oof_pred_bag',
                                   needs_proba=False, metric=accuracy_score, n_folds=4, stratified=True, shuffle=True, random_state=0, verbos

# Initialize and fit 2nd level model
final_model = XGBClassifier(random_state=123, n_jobs=-1, learning_rate=0.1, n_estimators=10, max_depth=3)
final_model_fit = final_model.fit(stack_train, y_train)

# Predict: stacked_pred
stacked_pred = final_model.predict(stack_test)

# Final prediction score
print('Final prediction score: [%.8f]' % accuracy_score(y_test, stacked_pred))
```

[1] https://towardsdatascience.com/automate [2] stacking [3] in [4] python [5] fc3e7834772e

Ensemble functions

Algorithm	Function
Bootstrap aggregation	`sklearn.ensemble.BaggingClassifier()`
Boosting	`sklearn.ensemble.AdaBoostClassifier()`
XGBoost	`xgboost.XGBClassifier()`

Bagging vs boosting

Technique	**Bias**	**Variance**
Bootstrap **agg**regation (Bagging)	Increase	**Decrease**
Boosting	**Decrease**	Increase

Major ensemble techniques MCQ

Which of the following statements is true about the three major techniques used for ensemble methods in Machine Learning? Select the statement that is **true**:

- Boosting methods decrease model variance.
- Boosting methods increase the predictive abilities of a classifier.
- Bootstrap aggregation, or bagging, decreases model bias.
- Model stacking takes the predictions from individual models and combines them to create a higher accuracy model.

Major ensemble techniques MCQ: answer

Which of the following statements is true about the three major techniques used for ensemble methods in Machine Learning? The correct answer is:

- **Model stacking takes the predictions from individual models and combines them to create a higher accuracy model.** (The final model obtained from the predictions of several individual models almost always outperforms the individuals.)

Major ensemble techniques MCQ: incorrect answers

Which of the following statements is true about the three major techniques used for ensemble methods in Machine Learning?

- Boosting methods decrease model variance. (Boosting methods decrease model bias which, at the same time, helps increase variance to find that sweet spot for best model generalization.)
- Boosting methods increase the predictive abilities of a classifier. (Boosting decreases model bias, which may or may not increase the predictive abilities of a classifier.)
- Bootstrap aggregation, or bagging, decreases model bias. (Bagging decreases model variance.)

Let's practice!

PREPARING FOR MACHINE LEARNING INTERVIEW QUESTIONS IN PYTHON