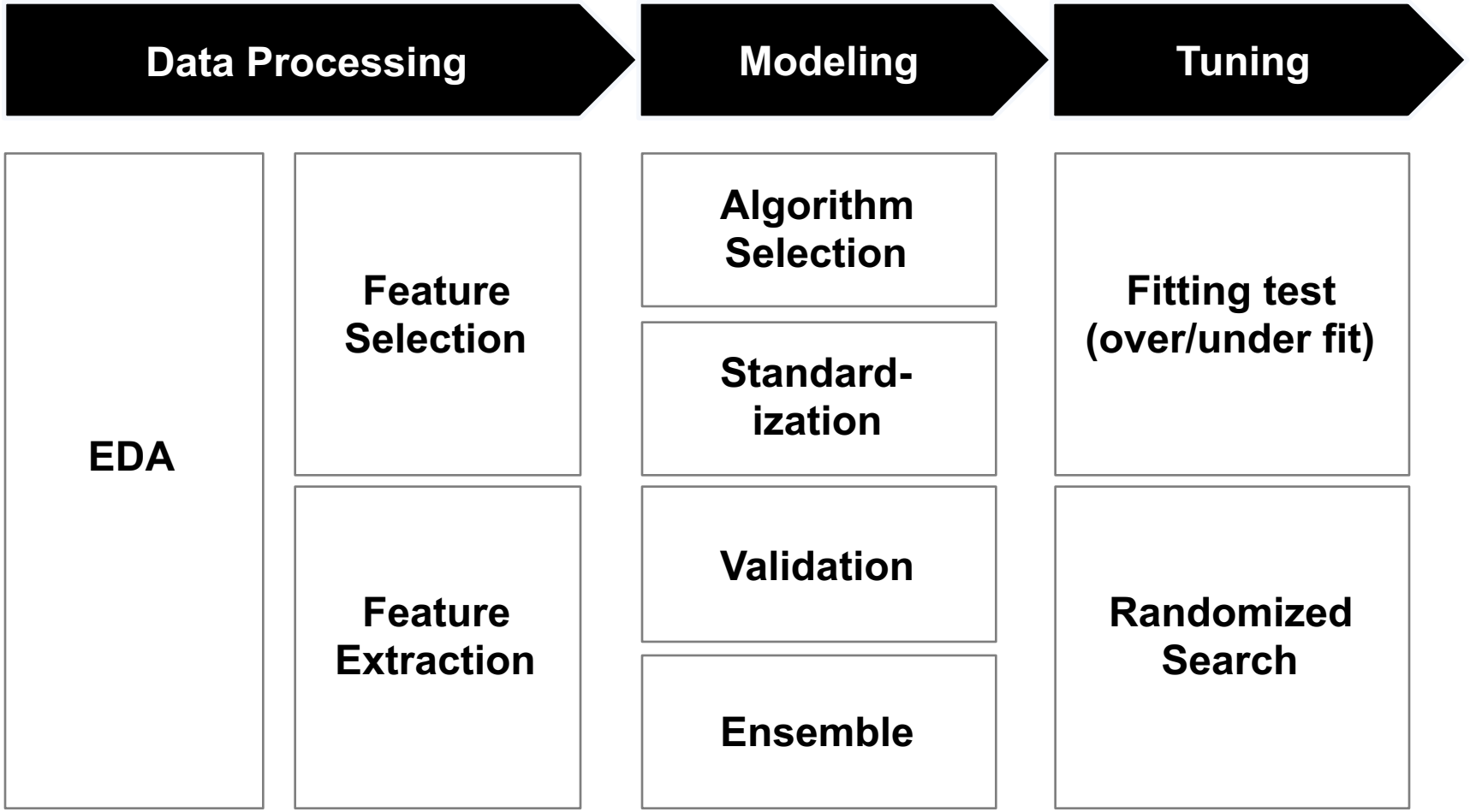
Churn Modeling with Telecom Data

Overview


Data Processing
Modeling
Tuning
EDA
Feature Selection
Feature Extraction
Algorithm Selection
Standard-ization
Validation
Ensemble
Fitting test (over/under fit)
Randomized Search

Data Processing

Feature selection and scatter matrix are used to figure out how to handle the features

	Original DF	DF1	DF2	DF3
# Features	21 features	71 features	21 features	17 features
Data Processing	• Drop phone # (but leave area code) • (Class is unbalanced)	Convert to dummy (n -1) • State • Intl plan • Vmail plan • Area code • Churn?	Feature selection • Drop State	Scatter matrix • Multi-collinearity between Charge and Min • Convert Vmail to 0 or 1

Data Processing: Feature Selection

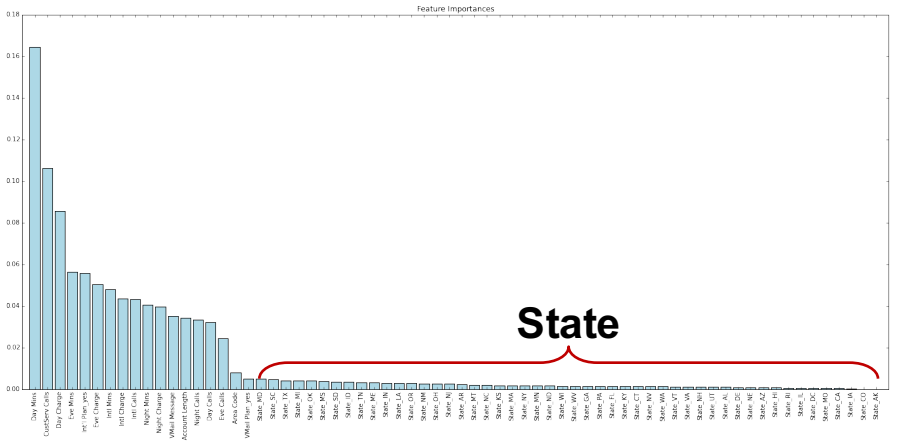
By looking at feature importance and optimal number of features, area information are not useful.

Features

DF2

21 features

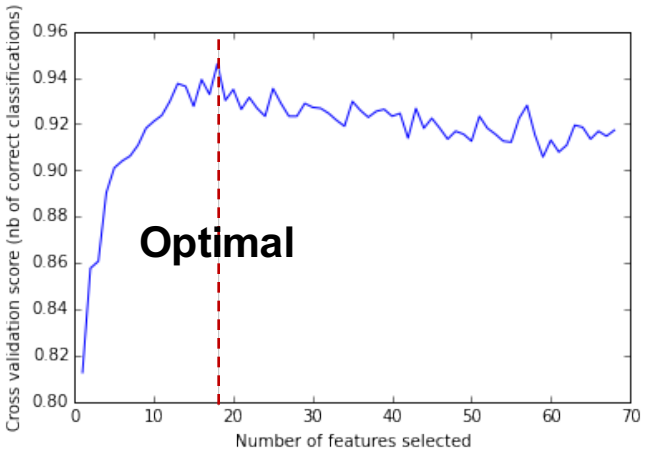
Feature Importance with Random Forest



Data Processing

Feature selection
- Drop State

Cross validated optimal number of features



- Number of DF1 = 71
- # of States = 51
- Optimal = 18 features

DF1 – State = close to optimal

Data Processing: Scatter Matrix

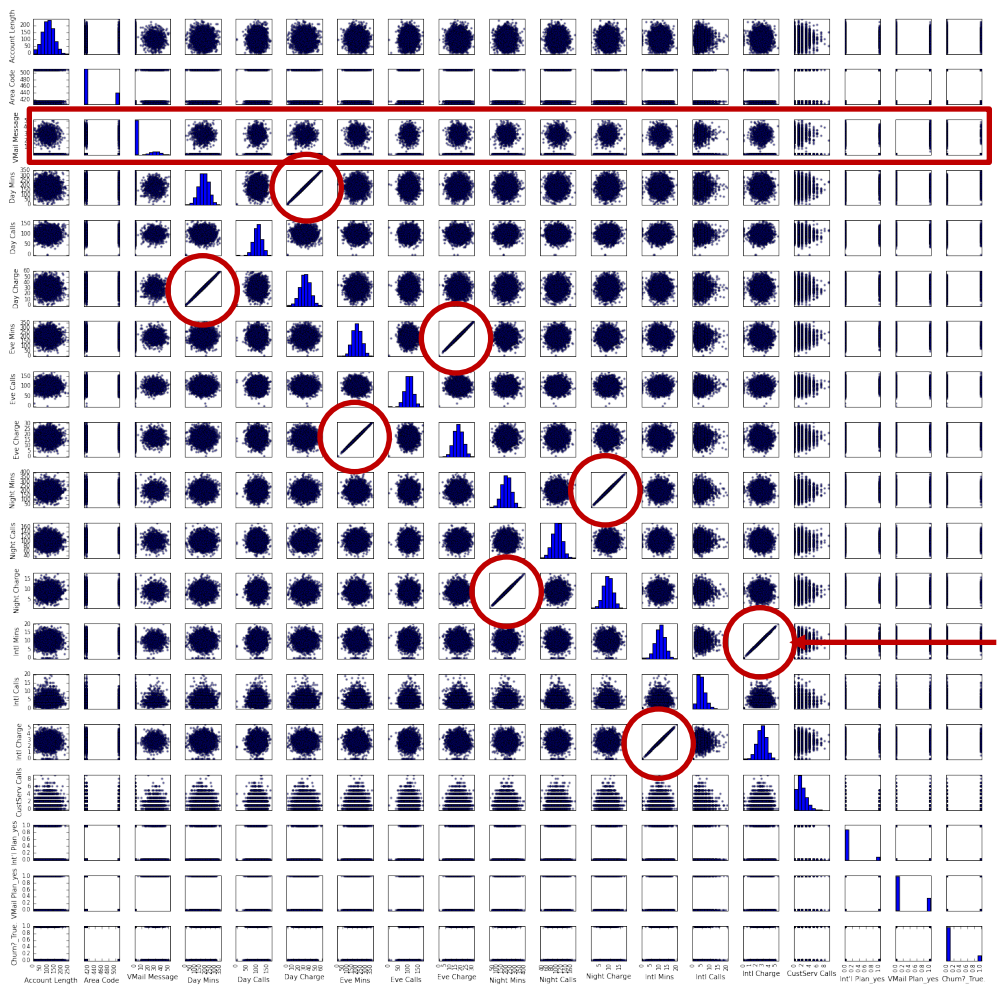
Two issues. (1) Multi-collinearity between Charge and Min (2) distribution of Vmail

DF3

Features

17 features

Scatter Matrix on DF3



Corr.=.99

Data Processing

Scatter matrix

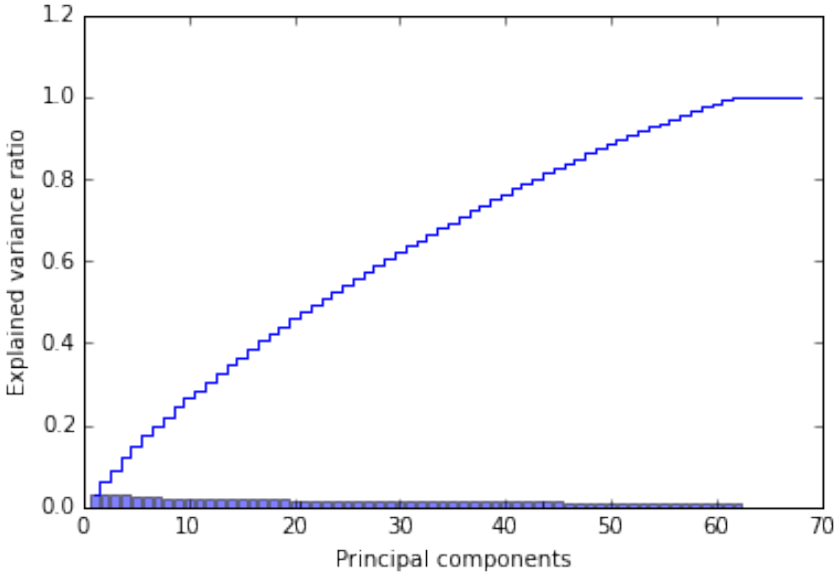
- Multi-collinearity between Charge and Min
- Convert Vmail to 0 or 1

Data Processing: Feature Extraction

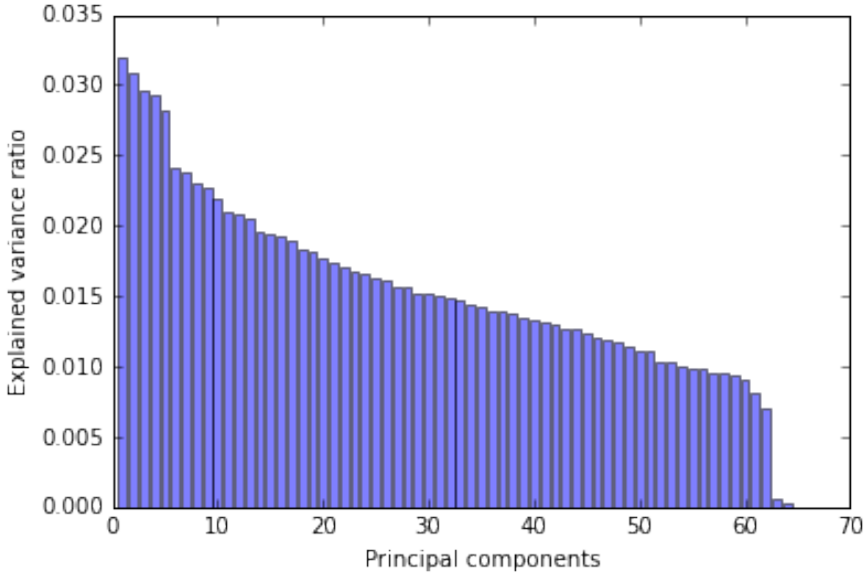
Since variances of principal components (PC) decrease smoothly, reducing dimensionality via PCA would not improve our model.

PCA and Explained Variance Ratio

With cumulated sum



Without cumulated sum



Modeling: Compare Algorithms

Top algorithms are gradient boosting and random forest.
Standardization doesn't improve the scores on tree-based algorithms.

Increase with Standardization | Decrease with Standardization | No change

	Accuracy	Accuracy w/ S.F.	Precision	Precision w/ S.F.	Recall	Recall w/ S.F.	F1	F1 w/ S.F.
Gradient Boosting	0.949	0.949	0.912	0.916	0.719	0.716	0.802	0.804
Random Forest	0.929	0.925	0.935	0.925	0.580	0.555	0.710	0.679
Decision Tree Balanced	0.919	0.920	0.718	0.712	0.727	0.725	0.715	0.711
Random Forest Balanced	0.919	0.919	0.929	0.931	0.478	0.492	0.662	0.631
Decision Tree	0.910	0.918	0.715	0.69	0.721	0.714	0.703	0.703
Ada Boost	0.876	0.876	0.623	0.623	0.356	0.356	0.451	0.451
KNN	0.868	0.879	0.603	0.738	0.240	0.253	0.34	0.375
Logistic Regression	0.860	0.862	0.573	0.567	0.166	0.215	0.252	0.310
SVM	0.855	0.913	0.00	0.872	0.00	0.468	0.00	0.607
Gaussian NB	0.854	0.854	0.497	0.497	0.432	0.432	0.461	0.461

Note: w/ S.F. = with standardized features
Note: cross_val_score(cv=10)

Modeling: Voting Classifier

Voting classifier of gradient boosting and random forest increases cross-validated scores

Gradient Boosting

- Accuracy: .949
- Precision: .912
- Recall: .719
- F1: .802

Random Forest

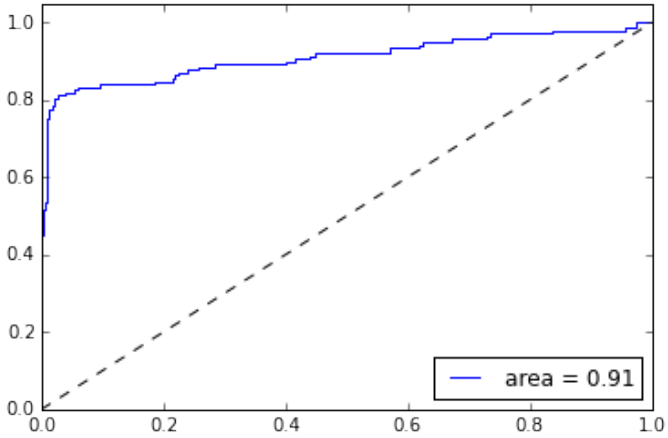
- Accuracy: .929
- Precision: .935
- Recall: .580
- F1: .710

Voting Classifier (VC)

Weight = 1(RF):2(GB)

- Accuracy: **.951**
- Precision: .926
- Recall: .716
- F1: **.805**

ROC (Non-CV)

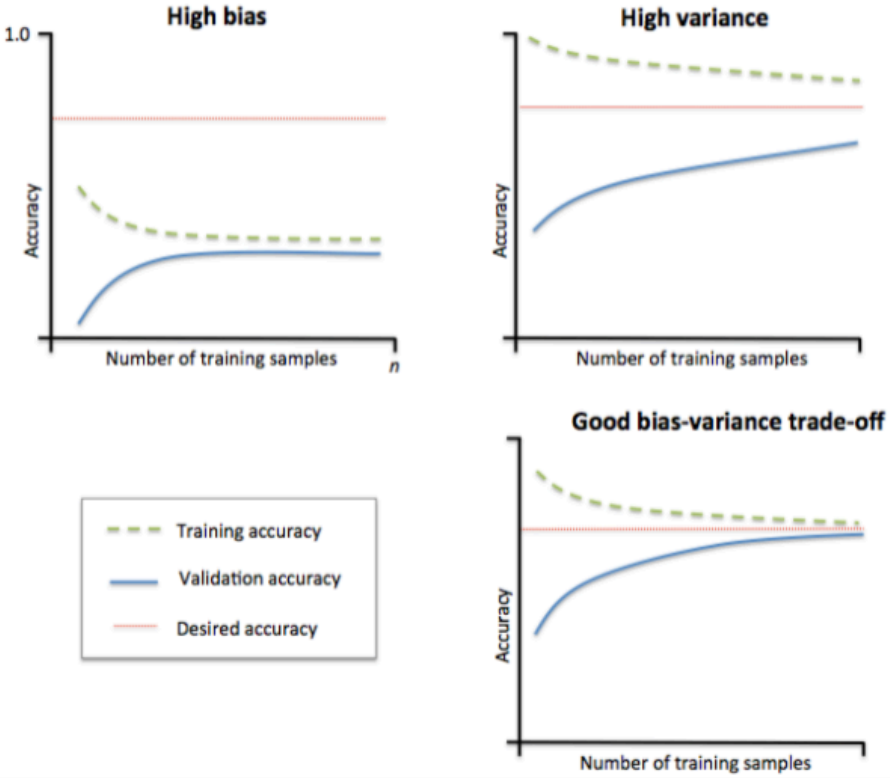


Note: cross_val_score(cv=10)

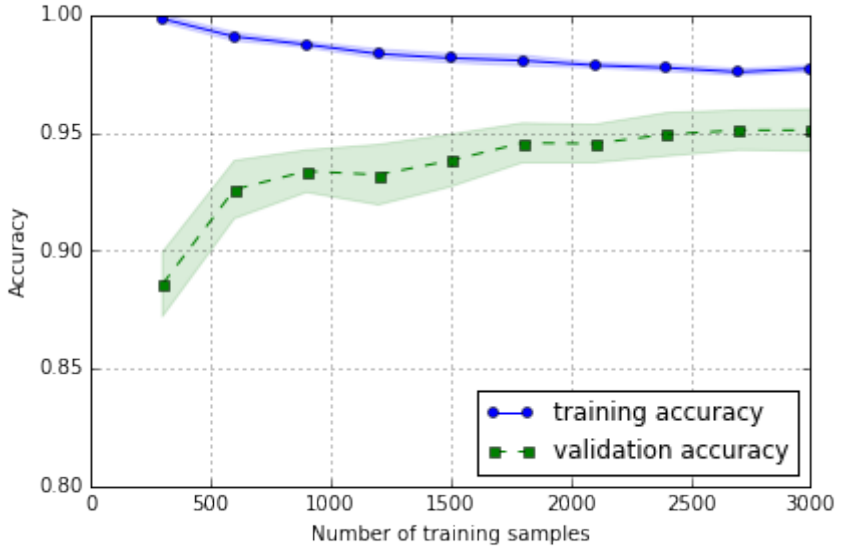
Tuning: Over/Under Fit

Learning curve shows that there is no major issue on fitting.

Example: Bias vs Variance



Learning Curve on VC



Low variance + Low bias

Source: Python Machine Learning

Tuning: Randomized Search

Optimizing parameters of voting classifier further increases accuracy measures.

Best Indiv. Algorithms → **Voting Classifier** → **Randomized Search**

Gradient Boosting

- Accuracy: .949
- Precision: .912
- Recall: .719
- F1: .802

Random Forest

- Accuracy: .929
- Precision: .935
- Recall: .580
- F1: .710

Voting Classifier

- Accuracy: .951
- Precision: .926
- Recall: .716
- F1: .805

Randomized Search VC

- Accuracy: **.956**
- Precision: .907
- Recall: .743
- F1: **.825**

Optimized Parameters:

```
RandomForestClassifier(n_jobs=-1, max_depth=3,
bootstrap=True, criterion='gini', max_features=4,
min_samples_split=3,
min_samples_leaf=8, warm_start=False, n_estimators
= 6070, class_weight='balanced_subsample' )

GradientBoostingClassifier(learning_rate=
0.3175974111166098, n_estimators=4798,
max_depth=None, max_features=5, warm_start=True,
min_samples_split=6, min_samples_leaf=5,
loss='exponential')
```

Note: cross_val_score(cv=10)

Summary

- **Preprocessing**
  - Feature importance to detect useful features
  - Cross-validated accuracy score to find optimal number of features
- **Model selection**
  - Tree algorithms wins (Gradient boosting classifier and random forest)
  - Standardization doesn't increase performance for tree algorithms unlike other algorithms
- **Voting classifier**
  - Voting classifier increases accuracy measures
- **Randomized search**
  - Randomized search increases accuracy measures