In [1]:

```
%matplotlib inline

import warnings
warnings.filterwarnings('ignore')

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.decomposition import PCA, KernelPCA
from sklearn.cross_validation import KFold, cross_val_score
from sklearn.metrics import make_scorer
from sklearn.grid_search import GridSearchCV
from sklearn.feature_selection import VarianceThreshold, RFE, SelectKBest, chi2, GenericUnivariateSelect,SelectFromModel
from sklearn.preprocessing import MinMaxScaler
from sklearn.pipeline import Pipeline, FeatureUnion
from sklearn.linear_model import LogisticRegression
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC,LinearSVC
from sklearn.svm import SVR
from sklearn.ensemble import BaggingClassifier, ExtraTreesClassifier, GradientBoostingClassifier, VotingClassifier, RandomForest
Classifier, AdaBoostClassifier
from sklearn import linear_model
sns.set_style('whitegrid')
pd.set_option('display.max_columns', None) # display all columns
```

```
/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.py:44: DeprecationWarning: This module was deprecated in version 0.
18 in favor of the model_selection module into which all the refactored classes and functions are moved. Also note that the interface
of the new CV iterators are different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
/usr/local/lib/python2.7/dist-packages/sklearn/grid_search.py:43: DeprecationWarning: This module was deprecated in version 0.18 in
favor of the model_selection module into which all the refactored classes and functions are moved. This module will be removed in 0.
20.
  DeprecationWarning)
```

In [2]:

```
data=pd.read_csv("/home/sonil/Documents/sonil/new bigniging/kaggle/data.csv")



data.set_index('shot_id', inplace=True)
data["action_type"] = data["action_type"].astype('object')
data["combined_shot_type"] = data["combined_shot_type"].astype('category')
data["game_event_id"] = data["game_event_id"].astype('category')
data["game_id"] = data["game_id"].astype('category')
data["period"] = data["period"].astype('object')
data["playoffs"] = data["playoffs"].astype('category')
data["season"] = data["season"].astype('category')
data["shot_made_flag"] = data["shot_made_flag"].astype('category')
data["shot_type"] = data["shot_type"].astype('category')
data["team_id"] = data["team_id"].astype('category')
```

In [3]:

```
data.head(2)
```

Out[3]:

	action_type	combined_shot_type	game_event_id	game_id	lat	loc_x	loc_y	lon	minutes_remain
shot_id									
1	Jump Shot	Jump Shot	10	20000012	33.9723	167	72	-118.1028	10

2	Jump Shot	Jump Shot	12	20000012	34.0443	-157	0	-118.4268	10

In [4]:
```
data.dtypes
```

Out[4]:
```
action_type             object
combined_shot_type    category
game_event_id         category
game_id               category
lat                    float64
loc_x                    int64
loc_y                    int64
lon                    float64
minutes_remaining        int64
period                  object
playoffs              category
season                category
seconds_remaining        int64
shot_distance            int64
shot_made_flag        category
shot_type             category
shot_zone_area          object
shot_zone_basic         object
shot_zone_range         object
team_id               category
team_name               object
game_date               object
matchup                 object
opponent                object
dtype: object
```

In [5]:
```
data.shape
```

Out[5]:
```
(30697, 24)
```

In [6]:
```
data.describe(include=['number'])
```

Out[6]:

	lat	loc_x	loc_y	lon	minutes_remaining	seconds_remaining	shot_distanc
count	30697.000000	30697.000000	30697.000000	30697.000000	30697.000000	30697.000000	30697.00000
mean	33.953192	7.110499	91.107535	-118.262690	4.885624	28.365085	13.437437
std	0.087791	110.124578	87.791361	0.110125	3.449897	17.478949	9.374189
min	33.253300	-250.000000	-44.000000	-118.519800	0.000000	0.000000	0.000000
25%	33.884300	-68.000000	4.000000	-118.337800	2.000000	13.000000	5.000000
50%	33.970300	0.000000	74.000000	-118.269800	5.000000	28.000000	15.000000
75%	34.040300	95.000000	160.000000	-118.174800	8.000000	43.000000	21.000000
max	34.088300	248.000000	791.000000	-118.021800	11.000000	59.000000	79.000000

In [7]:
```
data.describe(include=['object', 'category'])
```

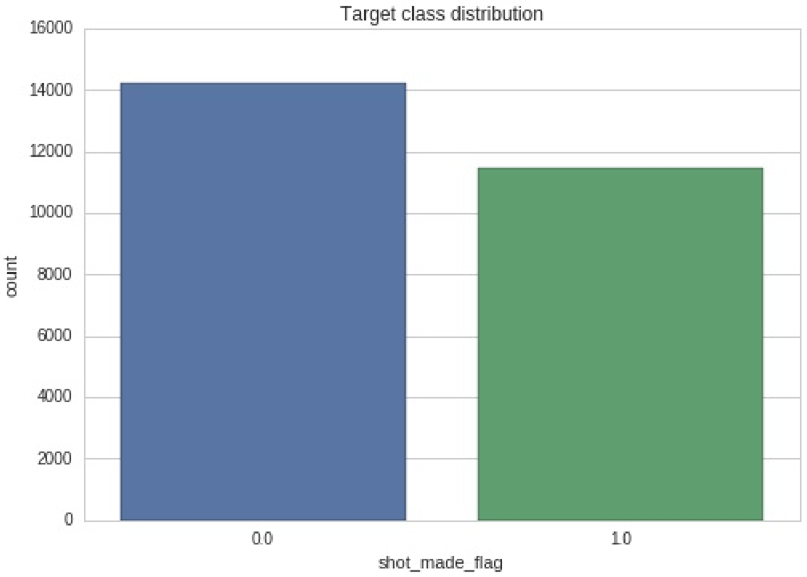
Out[7]:

	action_type	combined_shot_type	game_event_id	game_id	period	playoffs	season	shot_made_flag	shot_t
count	30697	30697	30697	30697	30697	30697	30697	25697.0	30697
unique	57	6	620	1559	7	2	20	2.0	2
top	Jump Shot	Jump Shot	2	21501228	3	0	2005- 06	0.0	2PT Fi Goal

freq	18880	23485	132	50	8296	26198	2318	14232.0	24271

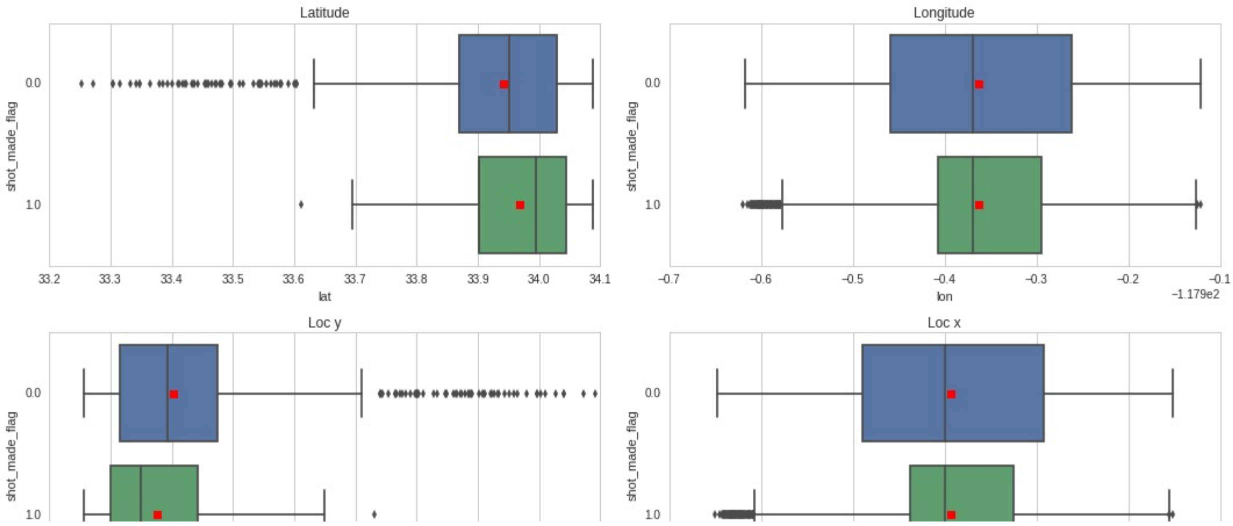
In [8]:

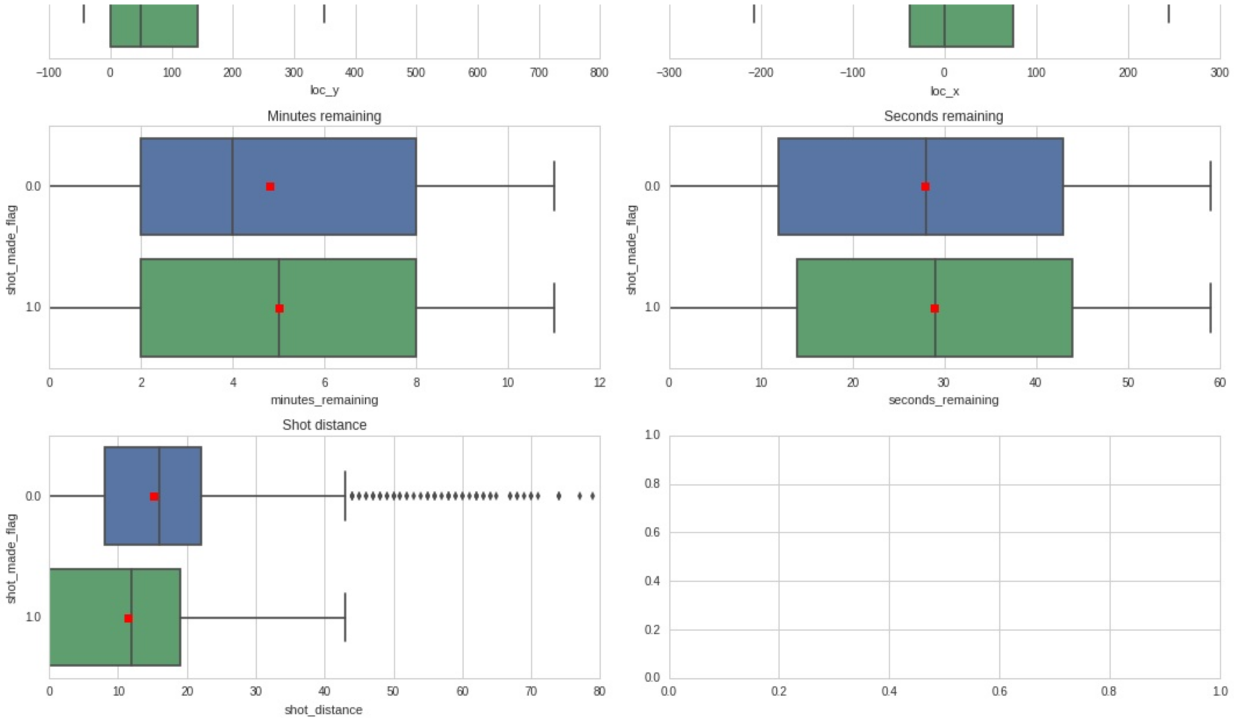
```
ax = plt.axes()
sns.countplot(x='shot_made_flag', data=data, ax=ax);
ax.set_title('Target class distribution')
plt.show()
```



In [9]:

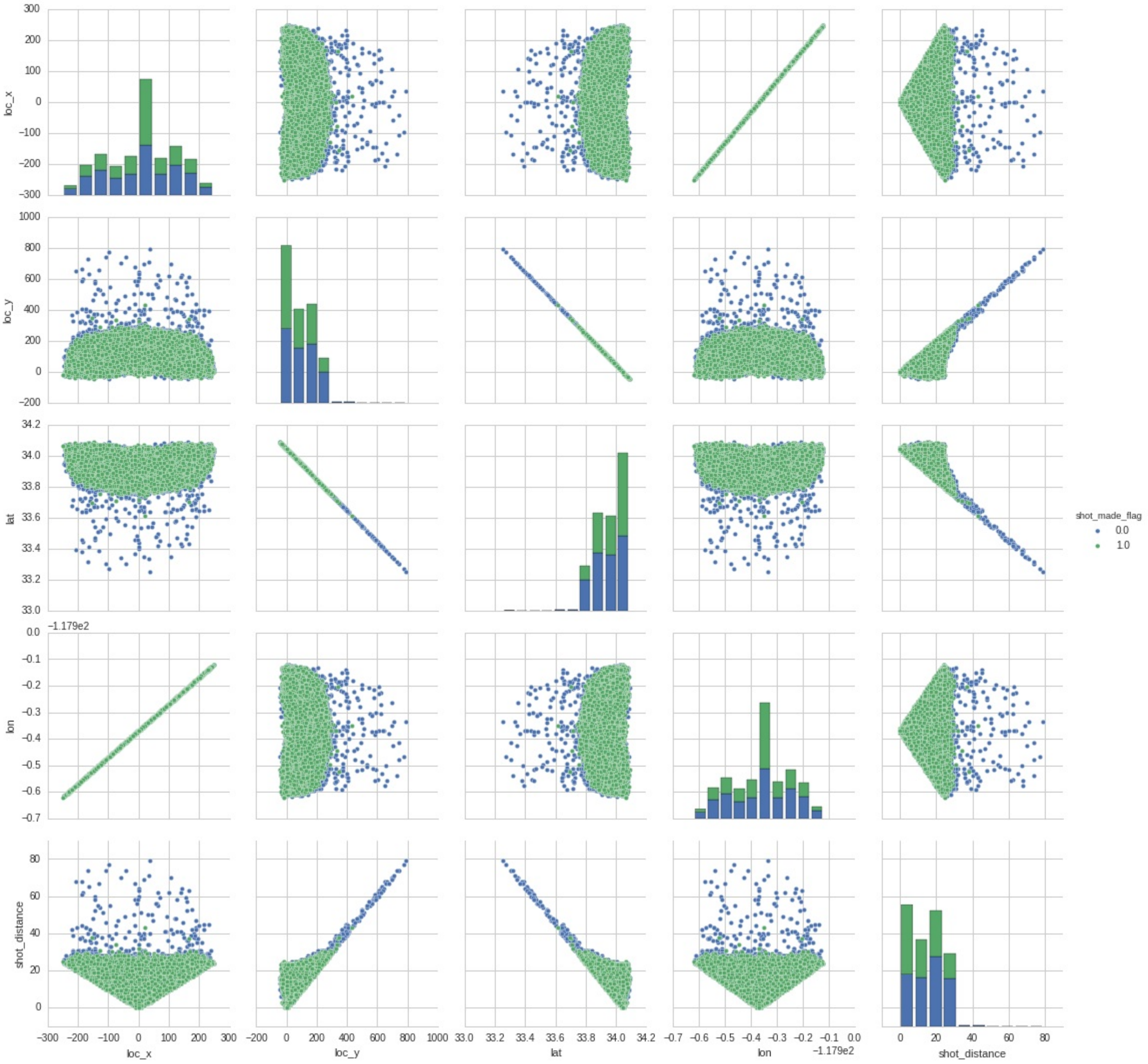
```
f, axarr = plt.subplots(4, 2, figsize=(15, 15))

sns.boxplot(x='lat', y='shot_made_flag', data=data, showmeans=True, ax=axarr[0,0])
sns.boxplot(x='lon', y='shot_made_flag', data=data, showmeans=True, ax=axarr[0, 1])
sns.boxplot(x='loc_y', y='shot_made_flag', data=data, showmeans=True, ax=axarr[1, 0])
sns.boxplot(x='loc_x', y='shot_made_flag', data=data, showmeans=True, ax=axarr[1, 1])
sns.boxplot(x='minutes_remaining', y='shot_made_flag', showmeans=True, data=data, ax=axarr[2, 0])
sns.boxplot(x='seconds_remaining', y='shot_made_flag', showmeans=True, data=data, ax=axarr[2, 1])
sns.boxplot(x='shot_distance', y='shot_made_flag', data=data, showmeans=True, ax=axarr[3, 0])

axarr[0, 0].set_title('Latitude')
axarr[0, 1].set_title('Longitude')
axarr[1, 0].set_title('Loc y')
axarr[1, 1].set_title('Loc x')
axarr[2, 0].set_title('Minutes remaining')
axarr[2, 1].set_title('Seconds remaining')
axarr[3, 0].set_title('Shot distance')

plt.tight_layout()
plt.show()
```





In [10]:

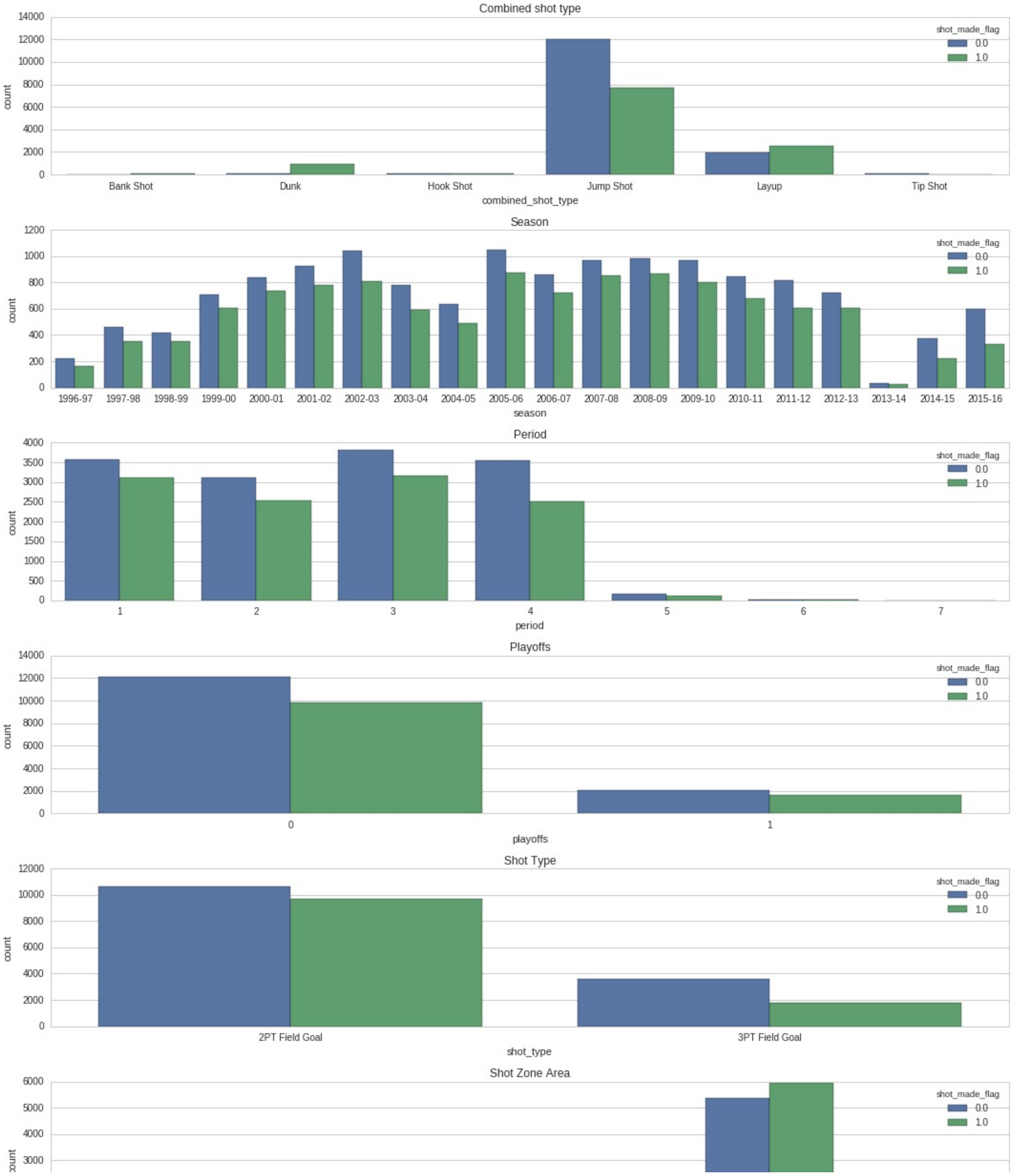
```
sns.pairplot(data, vars=['loc_x', 'loc_y', 'lat', 'lon', 'shot_distance'], hue='shot_made_flag', size=3)
plt.show()
```

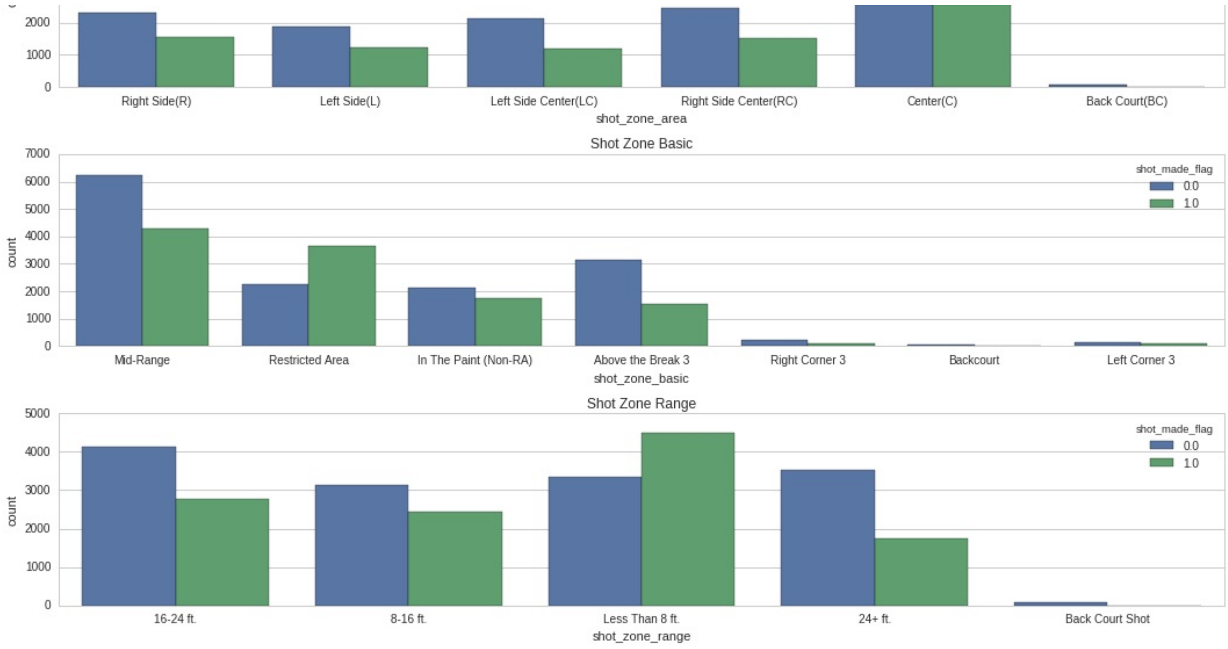


In [12]:

```
f, axarr = plt.subplots(8, figsize=(15, 25))
```

```
sns.countplot(x="combined_shot_type", hue="shot_made_flag", data=data, ax=axarr[0])
sns.countplot(x="season", hue="shot_made_flag", data=data, ax=axarr[1])
sns.countplot(x="period", hue="shot_made_flag", data=data, ax=axarr[2])
sns.countplot(x="playoffs", hue="shot_made_flag", data=data, ax=axarr[3])
sns.countplot(x="shot_type", hue="shot_made_flag", data=data, ax=axarr[4])
sns.countplot(x="shot_zone_area", hue="shot_made_flag", data=data, ax=axarr[5])
sns.countplot(x="shot_zone_basic", hue="shot_made_flag", data=data, ax=axarr[6])
sns.countplot(x="shot_zone_range", hue="shot_made_flag", data=data, ax=axarr[7])

axarr[0].set_title('Combined shot type')
axarr[1].set_title('Season')
axarr[2].set_title('Period')
axarr[3].set_title('Playoffs')
axarr[4].set_title('Shot Type')
axarr[5].set_title('Shot Zone Area')
axarr[6].set_title('Shot Zone Basic')
axarr[7].set_title('Shot Zone Range')

plt.tight_layout()
plt.show()
```





In [11]:

```
unknown_mask = data['shot_made_flag'].isnull()
```

In [12]:

```
data_cl = data.copy() # create a copy of data frame
target = data_cl['shot_made_flag'].copy()

# Remove some columns
data_cl.drop('team_id', axis=1, inplace=True) # Always one number
data_cl.drop('lat', axis=1, inplace=True) # Correlated with loc_x
data_cl.drop('lon', axis=1, inplace=True) # Correlated with loc_y
data_cl.drop('game_id', axis=1, inplace=True) # Independent
data_cl.drop('game_event_id', axis=1, inplace=True) # Independent
data_cl.drop('team_name', axis=1, inplace=True) # Always LA Lakers
data_cl.drop('shot_made_flag', axis=1, inplace=True)
```

In [13]:

```
def detect_outliers(series, whis=1.5):
    q75, q25 = np.percentile(series, [75 ,25])
    iqr = q75 - q25
    return ~((series - series.median()).abs() <= (whis * iqr))

## For now - do not remove anything
```

In [14]:

```
#Remaining time
data_cl['seconds_from_period_end'] = 60 * data_cl['minutes_remaining'] + data_cl['seconds_remaining']
data_cl['last_5_sec_in_period'] = data_cl['seconds_from_period_end'] < 5

data_cl.drop('minutes_remaining', axis=1, inplace=True)
data_cl.drop('seconds_remaining', axis=1, inplace=True)
data_cl.drop('seconds_from_period_end', axis=1, inplace=True)
## Matchup - (away/home)
data_cl['home_play'] = data_cl['matchup'].str.contains('vs').astype('int')
data_cl.drop('matchup', axis=1, inplace=True)
# Game date
data_cl['game_date'] = pd.to_datetime(data_cl['game_date'])
data_cl['game_year'] = data_cl['game_date'].dt.year
data_cl['game_month'] = data_cl['game_date'].dt.month
data_cl.drop('game_date', axis=1, inplace=True)

# Loc_x, and loc_y binning
data_cl['loc_x'] = pd.cut(data_cl['loc_x'], 25)
data_cl['loc_y'] = pd.cut(data_cl['loc_y'], 25)

# Replace 20 least common action types with value 'Other'
rare_action_types = data_cl['action_type'].value_counts().sort_values().index.values[:20]
```

```
data_cl.loc[data_cl['action_type'].isin(rare_action_types), 'action_type'] = 'Other'
```

In [15]:

```
categorial_cols = [
    'action_type', 'combined_shot_type', 'period', 'season', 'shot_type',
    'shot_zone_area', 'shot_zone_basic', 'shot_zone_range', 'game_year',
    'game_month', 'opponent', 'loc_x', 'loc_y']

for cc in categorial_cols:
    dummies = pd.get_dummies(data_cl[cc])
    dummies = dummies.add_prefix("{}#".format(cc))
    data_cl.drop(cc, axis=1, inplace=True)
    data_cl = data_cl.join(dummies)
```

In [16]:

```
# Separate dataset for validation
data_submit = data_cl[unknown_mask]

# Separate dataset for training
X = data_cl[~unknown_mask]
Y = target[~unknown_mask]
```

In [17]:

```
# feature selection using chisquare
model=SelectKBest(chi2,k=20)
features_chi2=data_cl.columns[model.fit(X,Y).get_support()]

features_chi2
```

Out[17]:

```
Index([u'shot_distance', u'last_5_sec_in_period',
       u'action_type#Driving Dunk Shot', u'action_type#Driving Layup Shot',
       u'action_type#Jump Bank Shot', u'action_type#Jump Shot',
       u'action_type#Pullup Jump shot', u'action_type#Running Jump Shot',
       u'action_type#Slam Dunk Shot', u'combined_shot_type#Dunk',
       u'combined_shot_type#Jump Shot', u'combined_shot_type#Layup',
       u'shot_type#3PT Field Goal', u'shot_zone_area#Center(C)',
       u'shot_zone_basic#Above the Break 3',
       u'shot_zone_basic#Restricted Area', u'shot_zone_range#24+ ft.',
       u'shot_zone_range#Less Than 8 ft.', u'loc_x#(-10.96, 8.96]',
       u'loc_y#(-10.6, 22.8]'],
      dtype='object')
```

In [18]:

```
#Feature Extraction using Learn from model using Linear SVM
lsvc = LinearSVC(C=0.01, penalty="l1", dual=False).fit(X, Y)
features = data_cl.columns[SelectFromModel(lsvc, prefit=True).get_support()]
features_selectFromModel_lSVC=features
features_selectFromModel_lSVC
```

Out[18]:

```
Index([u'shot_distance', u'last_5_sec_in_period', u'home_play',
       u'action_type#Driving Layup Shot', u'action_type#Dunk Shot',
       u'action_type#Fadeaway Jump Shot', u'action_type#Jump Bank Shot',
       u'action_type#Jump Shot', u'action_type#Layup Shot',
       u'action_type#Pullup Jump shot', u'action_type#Running Jump Shot',
       u'action_type#Turnaround Jump Shot', u'combined_shot_type#Dunk',
       u'combined_shot_type#Layup', u'combined_shot_type#Tip Shot',
       u'period#1', u'period#4', u'season#2000-01', u'season#2005-06',
       u'season#2006-07', u'season#2007-08', u'season#2008-09',
       u'season#2011-12', u'season#2014-15', u'season#2015-16',
       u'shot_zone_area#Center(C)', u'shot_zone_area#Right Side Center(RC)',
       u'shot_zone_basic#Restricted Area', u'shot_zone_range#16-24 ft.',
       u'game_year#2000', u'game_year#2006', u'game_year#2008',
       u'game_month#1', u'game_month#2', u'game_month#5', u'opponent#HOU',
       u'opponent#NYK', u'opponent#OKC', u'opponent#PHX', u'opponent#SAC',
       u'loc_x#(-130.48, -110.56]', u'loc_x#(-10.96, 8.96]',
       u'loc_y#(22.8, 56.2]', u'loc_y#(123, 156.4]', u'loc_y#(156.4, 189.8]',
       u'loc_y#(189.8, 223.2]'],
      dtype='object')
```

In [19]:

```
threshold = 0.90
vt = VarianceThreshold().fit(X)

# Find feature names
feat_var_threshold = data_cl.columns[vt.variances_ > threshold * (1-threshold)]
feat_var_threshold
```

Out[19]:

```
Index([u'playoffs', u'shot_distance', u'home_play', u'action_type#Jump Shot',
       u'combined_shot_type#Jump Shot', u'combined_shot_type#Layup',
       u'period#1', u'period#2', u'period#3', u'period#4',
       u'shot_type#2PT Field Goal', u'shot_type#3PT Field Goal',
       u'shot_zone_area#Center(C)', u'shot_zone_area#Left Side Center(LC)',
       u'shot_zone_area#Left Side(L)', u'shot_zone_area#Right Side Center(RC)',
       u'shot_zone_area#Right Side(R)', u'shot_zone_basic#Above the Break 3',
       u'shot_zone_basic#In The Paint (Non-RA)', u'shot_zone_basic#Mid-Range',
       u'shot_zone_basic#Restricted Area', u'shot_zone_range#16-24 ft.',
       u'shot_zone_range#24+ ft.', u'shot_zone_range#8-16 ft.',
       u'shot_zone_range#Less Than 8 ft.', u'game_month#1', u'game_month#2',
       u'game_month#3', u'game_month#4', u'game_month#11', u'game_month#12',
       u'loc_x#(-10.96, 8.96]', u'loc_y#(-10.6, 22.8]', u'loc_y#(22.8, 56.2]',
       u'loc_y#(123, 156.4]'],
      dtype='object')
```

In [20]:

```
model = RandomForestClassifier()
model.fit(X, Y)

feature_imp = pd.DataFrame(model.feature_importances_, index=X.columns, columns=["importance"])
feat_imp_20 = feature_imp.sort_values("importance", ascending=False).head(20).index
feat_imp_20
```

Out[20]:

```
Index([u'shot_distance', u'action_type#Jump Shot', u'home_play', u'period#3',
       u'period#2', u'period#1', u'combined_shot_type#Dunk',
       u'action_type#Layup Shot', u'period#4', u'game_month#1',
       u'game_month#3', u'game_month#4', u'game_month#12', u'game_month#11',
       u'game_month#2', u'action_type#Driving Layup Shot', u'playoffs',
       u'opponent#PHX', u'opponent#POR', u'opponent#DEN'],
      dtype='object')
```

In [22]:

```
rfe = RFE(LogisticRegression(), 20)
rfe.fit(X, Y)

feature_rfe_scoring = pd.DataFrame({
        'feature': X.columns,
        'score': rfe.ranking_
    })

feat_rfe_20 = feature_rfe_scoring[feature_rfe_scoring['score'] == 1]['feature'].values
feat_rfe_20
```

Out[22]:

```
array(['action_type#Driving Dunk Shot',
       'action_type#Driving Finger Roll Layup Shot',
       'action_type#Driving Finger Roll Shot',
       'action_type#Driving Slam Dunk Shot', 'action_type#Dunk Shot',
       'action_type#Fadeaway Bank shot', 'action_type#Finger Roll Shot',
       'action_type#Hook Shot', 'action_type#Jump Shot',
       'action_type#Layup Shot', 'action_type#Running Bank shot',
       'action_type#Running Hook Shot', 'action_type#Slam Dunk Shot',
       'combined_shot_type#Dunk', 'combined_shot_type#Tip Shot',
       'shot_zone_area#Back Court(BC)', 'shot_zone_range#Back Court Shot',
       'loc_y#(290, 323.4]', 'loc_y#(356.8, 390.2]', 'loc_y#(390.2, 423.6]'], dtype=object)
```

Out[22]:

```
array(['action_type#Driving Dunk Shot',
       'action_type#Driving Finger Roll Layup Shot',
       'action_type#Driving Finger Roll Shot',
       'action_type#Driving Slam Dunk Shot', 'action_type#Dunk Shot',
       'action_type#Fadeaway Bank shot', 'action_type#Finger Roll Shot',
       'action_type#Hook Shot', 'action_type#Jump Shot',
       'action_type#Layup Shot', 'action_type#Running Bank shot',
       'action_type#Running Hook Shot', 'action_type#Slam Dunk Shot',
       'combined_shot_type#Dunk', 'combined_shot_type#Tip Shot',
       'shot_zone_area#Back Court(BC)', 'shot_zone_range#Back Court Shot',
       'loc_y#(290, 323.4]', 'loc_y#(356.8, 390.2]', 'loc_y#(390.2, 423.6]'], dtype=object)
```

In [23]:

```
#clf = linear_model.Lasso(alpha=0.001)
#clf.fit(X,Y)
#print(clf.coef_)
#print(clf.intercept_)

num_instances=len(X)
num_folds=3
kfold=KFold(n=num_instances, n_folds=num_folds)
model=linear_model.Lasso(alpha=0.001)

cv_results = cross_val_score(model, X, Y, cv=kfold, scoring='neg_mean_absolute_error', n_jobs=1)
print cv_results
print model.coef_
```

```
[-0.42475067 -0.43210373 -0.42578601]
None
```

In [24]:

```
#feature extraction using Lasso

#alpha=[1,0.1,0.01,0.001,.0001,.00001,.000001]
#nonZero=[]
#rSquare=[]

#for a in alpha:

 #   model=linear_model.Lasso(alpha=a, fit_intercept=True)
  #  model.fit(X,Y)
  #  nonZero.append(np.count_nonzero(model.coef_))
  #  rSquare.append(model.score(X,Y))

model = linear_model.Lasso(alpha=.001)
model.fit(X, Y)

feature_imp = pd.DataFrame(model.coef_, index=X.columns, columns=["importance"])
feat_imp_lasso = feature_imp.sort_values("importance", ascending=False).head(20).index
feat_imp_lasso
```

Out[24]:

```
Index([u'combined_shot_type#Dunk', u'action_type#Driving Layup Shot',
       u'action_type#Running Jump Shot', u'loc_y#(123, 156.4]',
       u'action_type#Jump Bank Shot', u'loc_y#(156.4, 189.8]',
       u'game_year#2006', u'shot_zone_range#16-24 ft.', u'game_year#2000',
       u'loc_x#(-10.96, 8.96]', u'shot_zone_area#Right Side Center(RC)',
       u'opponent#PHX', u'period#1', u'loc_y#(189.8, 223.2]',
       u'shot_zone_area#Center(C)', u'home_play', u'shot_zone_range#24+ ft.',
       u'opponent#SAC', u'game_month#5', u'season#2005-06'],
      dtype='object')
```

In [25]:

```
print('Clean dataset shape: {}'.format(data_cl.shape))
print('Subbmitable dataset shape: {}'.format(data_submit.shape))
print('Train features shape: {}'.format(X.shape))
print('Target label shape: {}'. format(Y.shape))
```

```
Clean dataset shape: (30697, 208)
Subbmitable dataset shape: (5000, 208)
Train features shape: (25697, 208)
Target label shape: (25697,)
```

In [26]:

```
# running model taking feature extraction one at a time
```

In [26]:

```
# running model taking feature extraction one at a time
features=[]

features.append(('variance',feat_var_threshold))
features.append(('rf',feat_imp_20))
features.append(('rfe',feat_rfe_20))
features.append(('lSVC',features_selectFromModel_lSVC))
features.append(('lasso',feat_imp_lasso))
```

In [27]:

```
#preparing model lists
models = []
models.append(('lr',LogisticRegression()))
models.append(('lda',LinearDiscriminantAnalysis()))
models.append(('CART',DecisionTreeClassifier()))
models.append(('rf',RandomForestClassifier()))

kfold=KFold(len(X),n_folds=3)
```

In [28]:

```
#running different models
for fname,feature in features:
    for name, model in models:
        cross=cross_val_score(model,X[feature],Y,scoring='log_loss',cv=kfold,n_jobs=1)
        print fname,name,cross.mean()
```

```
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

variance lr -0.632762527016
variance
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 lda -0.633358567873
variance
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
 CART -9.23046199092
variance
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 rf -2.03110393785
rf
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 lr -0.620378750969
rf
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 lda -0.621770160147
rf
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n

eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 CART -5.48109901628
rf
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 rf -1.84642312798
rfe
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 lr -0.615249011606
rfe
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 lda -0.617249521855
rfe
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 CART -0.626168406187
rfe
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 rf -0.626297081844
lSVC
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 lr -0.612435293573
lSVC
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
 lda -0.615328743564
lSVC
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 CART -9.90377302107
lSVC
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 rf -1.65046719666
lasso
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 lr -0.640529215854
lasso
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 lda -0.642758624566
lasso
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

 CART -1.90453588303
lasso rf -0.914473684581
```

In [29]:

```
features1 = np.hstack([
        feat_var_threshold,
        feat_imp_20,
        feat_rfe_20
    ])

features1 = np.unique(features1)



features2 = np.hstack([
        feat_var_threshold,
        feat_imp_20,
        feat_rfe_20,
        features_chi2,
        features_selectFromModel_lSVC,
        feat_imp_lasso
    ])

features2 = np.unique(features2)

print('Final features set:\n')
#for f in features:
    #print("\t-{}".format(f))
```

```
Final features set:
```

In [40]:

```
models = []
#models.append(('lr',LogisticRegression()))
models.append(('xg',xgboost.XGBClassifier()))
```

In [36]:

```
#test different models with feature set1 and feature set2
features=[]
features.append(('oldFeatures',features1))
features.append(('newFeatures',features2))
```

In [41]:

```
for ftype,feature in features:
    for name, model in models:
        cross=cross_val_score(model,X[feature],Y,scoring='log_loss',cv=kfold,n_jobs=1)
        print ftype,name,cross.mean()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-41-124a99e0a29c> in <module>()
      2 for ftype,feature in features:
      3     for name, model in models:
----> 4         cross=cross_val_score(model,X[feature],Y,scoring='log_loss',cv=kfold,n_jobs=1)
      5         print ftype,name,cross.mean()

/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.pyc in cross_val_score(estimator, X, y, scoring, cv, n_jobs, verbose, f
it_params, pre_dispatch)
   1569                                               train, test, verbose, None,
   1570                                               fit_params)
-> 1571                       for train, test in cv)
   1572     return np.array(scores)[:, 0]
   1573 

/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/parallel.pyc in __call__(self, iterable)
    756             # was dispatched. In particular this covers the edge
    757             # case of Parallel used with an exhausted iterator.
--> 758             while self.dispatch_one_batch(iterator):
    759                 self._iterating = True
    760             else:
```

```
/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/parallel.pyc in dispatch_one_batch(self, iterator)
    606                 return False
    607             else:
--> 608                 self._dispatch(tasks)
    609                 return True
    610 

/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/parallel.pyc in _dispatch(self, batch)
    569         dispatch_timestamp = time.time()
    570         cb = BatchCompletionCallBack(dispatch_timestamp, len(batch), self)
--> 571         job = self._backend.apply_async(batch, callback=cb)
    572         self._jobs.append(job)
    573 

/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/_parallel_backends.pyc in apply_async(self, func, callback)
    107     def apply_async(self, func, callback=None):
    108         """Schedule a func to be run"""
--> 109         result = ImmediateResult(func)
    110         if callback:
    111             callback(result)

/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/_parallel_backends.pyc in __init__(self, batch)
    324         # Don't delay the application, to avoid keeping the input
    325         # arguments in memory
--> 326         self.results = batch()
    327 
    328     def get(self):

/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/parallel.pyc in __call__(self)
    129 
    130     def __call__(self):
--> 131         return [func(*args, **kwargs) for func, args, kwargs in self.items]
    132 
    133     def __len__(self):

/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.pyc in _fit_and_score(estimator, X, y, scorer, train, test, verbose, para
meters, fit_params, return_train_score, return_parameters, error_score)
   1663             estimator.fit(X_train, **fit_params)
   1664         else:
-> 1665             estimator.fit(X_train, y_train, **fit_params)
   1666 
   1667     except Exception as e:

/usr/local/lib/python2.7/dist-packages/xgboost/sklearn.pyc in fit(self, X, y, sample_weight, eval_set, eval_metric, early_stopping_roun
ds, verbose)
    437         else:
    438             train_dmatrix = DMatrix(X, label=training_labels,
--> 439                                     missing=self.missing)
    440 
    441         self._Booster = train(xgb_options, train_dmatrix, self.n_estimators,

/usr/local/lib/python2.7/dist-packages/xgboost/core.pyc in __init__(self, data, label, missing, weight, silent, feature_names, feature_ty
pes)
    253         data, feature_names, feature_types = _maybe_pandas_data(data,
    254                                                                 feature_names,
--> 255                                                                 feature_types)
    256         label = _maybe_pandas_label(label)
    257 

/usr/local/lib/python2.7/dist-packages/xgboost/core.pyc in _maybe_pandas_data(data, feature_names, feature_types)
    179         msg = """DataFrame.dtypes for data must be int, float or bool.
    180 Did not expect the data types in fields """
--> 181         raise ValueError(msg + ', '.join(bad_fields))
    182 
    183     if feature_names is None:

ValueError: DataFrame.dtypes for data must be int, float or bool.
Did not expect the data types in fields playoffs
```

In [43]:

```
model = xgboost.XGBClassifier()
model.fit(X[feature], Y)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)

<ipython-input-43-27b37836fd64> in <module>()
      1 model = xgboost.XGBClassifier()
----> 2 model.fit(X[feature], Y)

/usr/local/lib/python2.7/dist-packages/xgboost/sklearn.pyc in fit(self, X, y, sample_weight, eval_set, eval_metric, early_stopping_roun
ds, verbose)
    437         else:
    438             train_dmatrix = DMatrix(X, label=training_labels,
--> 439                                     missing=self.missing)
    440 
    441         self._Booster = train(xgb_options, train_dmatrix, self.n_estimators,

/usr/local/lib/python2.7/dist-packages/xgboost/core.pyc in __init__(self, data, label, missing, weight, silent, feature_names, feature_ty
pes)
    253         data, feature_names, feature_types = _maybe_pandas_data(data,
    254                                                                 feature_names,
--> 255                                                                 feature_types)
    256         label = _maybe_pandas_label(label)
    257 

/usr/local/lib/python2.7/dist-packages/xgboost/core.pyc in _maybe_pandas_data(data, feature_names, feature_types)
    179         msg = """DataFrame.dtypes for data must be int, float or bool.
    180 Did not expect the data types in fields """
--> 181         raise ValueError(msg + ', '.join(bad_fields))
    182 
    183     if feature_names is None:

ValueError: DataFrame.dtypes for data must be int, float or bool.
Did not expect the data types in fields playoffs
Exception AttributeError: "'DMatrix' object has no attribute 'handle'" in <bound method DMatrix.__del__ of <xgboost.core.DMatrix obje
ct at 0x7f7e59f89210>> ignored
```

In [46]:

```
X[feature].feature_names()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-46-6d89385d8e98> in <module>()
----> 1 X[feature].feature_names()

/usr/local/lib/python2.7/dist-packages/pandas/core/generic.pyc in __getattr__(self, name)
   2742             if name in self._info_axis:
   2743                 return self[name]
-> 2744             return object.__getattribute__(self, name)
   2745 
   2746     def __setattr__(self, name, value):

AttributeError: 'DataFrame' object has no attribute 'feature_names'
```

In [145]:

```
#try both featureset with different models
#1. Bagging
cart = DecisionTreeClassifier()
num_trees = 100
scoring='log_loss'
```

```
processors=1

model = BaggingClassifier(base_estimator=cart, n_estimators=num_trees)

for ftype,feature in features:
    results = cross_val_score(model, X[feature], Y, cv=kfold, scoring=scoring, n_jobs=processors)
    print ftype,
    print(": ({0:.3f}) +/- ({1:.3f})".format(results.mean(), results.std()))
```

```
oldFeatures : (-0.876) +/- (0.025)
newFeatures : (-0.766) +/- (0.034)
```

In [42]:

```
#2. Random Forest
num_trees = 100
num_features = 10

model = RandomForestClassifier(n_estimators=num_trees, max_features=num_features)

for ftype,feature in features:
    results = cross_val_score(model, X[feature], Y, cv=kfold, scoring=scoring, n_jobs=processors)
    print ftype,
    print("({0:.3f}) +/- ({1:.3f})".format(results.mean(), results.std()))
```

```
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)
/usr/local/lib/python2.7/dist-packages/sklearn/metrics/scorer.py:127: DeprecationWarning: Scoring method log_loss was renamed to n
eg_log_loss in version 0.18 and will be removed in 0.20.
  sample_weight=sample_weight)

oldFeatures (-0.894) +/- (0.043)
newFeatures (-0.733) +/- (0.030)
```

In [147]:

```
#3. Ada Boosting
model = AdaBoostClassifier(n_estimators=100)
for ftype,feature in features:
    results = cross_val_score(model, X, Y, cv=kfold, scoring=scoring, n_jobs=processors)
    print ftype,
    print("({0:.3f}) +/- ({1:.3f})".format(results.mean(), results.std()))
```

```
oldFeatures (-0.690) +/- (0.000)
newFeatures (-0.690) +/- (0.000)
```

In [149]:

```
#4.Stochastic Gradient Boosting
model = GradientBoostingClassifier(n_estimators=100)
for ftype,feature in features:
    results = cross_val_score(model, X, Y, cv=kfold, scoring=scoring, n_jobs=processors)
    print ftype,
    print("({0:.3f}) +/- ({1:.3f})".format(results.mean(), results.std()))
```

```
oldFeatures (-0.609) +/- (0.002)
newFeatures (-0.609) +/- (0.002)
```

In [151]:

```
#Hyper paramter tuning
#1. logistics Regression
lr_grid = GridSearchCV(
    estimator = LogisticRegression(),
    param_grid = {
        'penalty': ['l1', 'l2'],
        'C': [0.001, 0.01, 1, 10, 100, 1000]
    },
    cv = kfold,
    scoring = scoring,
    n_jobs = processors)

lr_grid.fit(X[features2], Y)

print(lr_grid.best_score_)
print(lr_grid.best_params_)
```

```
-0.609891727148
{'penalty': 'l1', 'C': 1}
```

In [152]:

```
#2. LDA
lda_grid = GridSearchCV(
    estimator = LinearDiscriminantAnalysis(),
    param_grid = {
        'solver': ['lsqr'],
        'shrinkage': [0, 0.25, 0.5, 0.75, 1],
        'n_components': [None, 2, 5, 10]
    },
    cv = kfold,
    scoring = scoring,
    n_jobs = processors)

lda_grid.fit(X[features2], Y)

print(lda_grid.best_score_)
print(lda_grid.best_params_)
```

```
-0.612883118738
{'shrinkage': 0, 'n_components': None, 'solver': 'lsqr'}
```

In [155]:

```
rf_grid = GridSearchCV(
    estimator = RandomForestClassifier(warm_start=True),
    param_grid = {
        'n_estimators': [100, 200],
        'criterion': ['gini', 'entropy'],
        'max_features': [18, 20],
        'max_depth': [8, 10],
        'bootstrap': [True]
    },
    cv = kfold,
    scoring = scoring,
    n_jobs = processors)

rf_grid.fit(X[features2], Y)

print(rf_grid.best_score_)
print(rf_grid.best_params_)
```

```
-0.606921194234
{'max_features': 20, 'n_estimators': 100, 'bootstrap': True, 'criterion': 'entropy', 'max_depth': 10}
```

In [157]:

```
ada_grid = GridSearchCV(
    estimator = AdaBoostClassifier(),
    param_grid = {
        'algorithm': ['SAMME', 'SAMME.R'],
        'n_estimators': [10, 25, 50],
        'learning_rate': [1e-3, 1e-2, 1e-1]
    },
    cv = kfold,
    scoring = scoring,
    n_jobs = processors)

ada_grid.fit(X[features2], Y)

print(ada_grid.best_score_)
print(ada_grid.best_params_)
```

```
-0.640973140848
{'n_estimators': 10, 'learning_rate': 0.001, 'algorithm': 'SAMME.R'}
```

In [158]:

```
gbm_grid = GridSearchCV(
    estimator = GradientBoostingClassifier(warm_start=True),
    param_grid = {
        'n_estimators': [100, 200],
        'max_depth': [2, 3, 4],
        'max_features': [10, 15, 20],
        'learning_rate': [1e-1, 1]
    },
    cv = kfold,
    scoring = scoring,
    n_jobs = processors)

gbm_grid.fit(X[features2], Y)

print(gbm_grid.best_score_)
print(gbm_grid.best_params_)
```

```
-0.606186187808
{'max_features': 15, 'n_estimators': 200, 'learning_rate': 0.1, 'max_depth': 3}
```

In [166]:

```
# Create sub models
estimators = []

estimators.append(('lr', LogisticRegression(penalty='l1', C=1)))
estimators.append(('lda',LinearDiscriminantAnalysis(shrinkage= 0, n_components= None, solver= 'lsqr')))
estimators.append(('gbm', GradientBoostingClassifier(n_estimators=200, max_depth=3, learning_rate=0.1, max_features=15, warm_st
art=True)))
estimators.append(('rf', RandomForestClassifier(bootstrap=True, max_depth=10, n_estimators=100, max_features=20, criterion='entro
py')))
estimators.append(('ada', AdaBoostClassifier(algorithm='SAMME.R', learning_rate=.001, n_estimators=10)))

# create the ensemble model
ensemble = VotingClassifier(estimators, voting='soft', weights=[1,2,1,3,1])

results = cross_val_score(ensemble, X, Y, cv=kfold, scoring=scoring,n_jobs=1)
print("({0:.3f}) +/- ({1:.3f})".format(results.mean(), results.std()))
```

```
(-0.611) +/- (0.002)
```

In [168]:

```
model = ensemble

model.fit(X, Y)
preds = model.predict_proba(data_submit)

submission = pd.DataFrame()
submission["shot_id"] = data_submit.index
```

```
submission["shot_made_flag"]= preds[:,0]

submission.to_csv("sub.csv",index=False)
```

In []:

```
```