Executable Jupyter notebook 3: Medical data for classification

In [1]:
```
# imports and plotting utility functions
%matplotlib inline
import warnings

import numpy as np
import pandas as pd

from sklearn.datasets import make_regression
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.ensemble import RandomForestClassifier
from sklearn.cross_validation import ShuffleSplit
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import average_precision_score

import seaborn as sns
from matplotlib import pylab as plt
from statsmodels.discrete.discrete_model import Logit
from scipy.linalg import norm

warnings.simplefilter('ignore')

rf_cmp = RandomForestClassifier(n_estimators=250, bootstrap=True, oob_
score=True, random_state=0)

def plot_lr(true_coefs, est_coefs, pvals, var_names=None, rf_cmp_coef=
None):
    n_feat = len(est_coefs)
    where_sign = lr_pvalues < 0.05
    plt.figure(figsize=(17, 6))
    # print non-significant betas
    plt.scatter(np.arange(X.shape[1]), est_coefs, s=150, color='red',
label='estimated betas', alpha=0.5)
    if true_coefs is not None:
        plt.scatter(np.arange(X.shape[1]), true_coefs, s=150, color='b
lack', label='true betas', alpha=0.5)
    if rf_cmp_coef is not None:
        plt.scatter(np.arange(X.shape[1]), rf_cmp_coef, s=150, marker=
'D', color='steelblue', label='RandomForest importances', alpha=0.5)
```

```
    # print star significant betas and their values
    axes = plt.gca()
    #import pdb; pdb.set_trace()
    y_min, y_max = axes.get_ylim()
    axes.set_ylim(y_min * 1.25, y_max * 1.25)
    sign_y = np.sum(where_sign) * [y_min]
    plt.scatter(np.arange(X.shape[1])[where_sign], sign_y, color='red'
, label='significant at p<0.05', s=150, marker=(5, 1), alpha=0.75, lin
ewidth=3)
    for i_b, p in enumerate(pvals):
        plt.text(x=i_b - 0.25, y=y_min * 1.10, s='$p$=%.3f' % p)

    plt.xlabel('input variables')
    if var_names is None:
        plt.xticks(np.arange(n_feat), (np.arange(n_feat) + 1), fontsiz
e=15)
    else:
        plt.xticks(np.arange(n_feat), var_names, fontsize=12, rotation
=90)
    plt.grid(True)
    plt.title('Logistic regression', fontsize=16)
    plt.legend(loc='upper right', fontsize=14, fancybox=True, framealp
ha=0.5)

def plot_regr_paths(coefs, accs, nonzeros, C_grid, var_names=None, unb
iased_accs=None):
    n_cols = 2
    n_rows = 1
    n_verticals = len(coefs)
    n_feat = len(coefs)

    my_palette = np.array([
        '#F47D7D', '#FBEF69', '#98E466', '#000000',
        '#A7794F', '#CCCCCC', '#85359C', '#FF9300', '#FF0030', 'grey',
'blue', 'salmon', '#4BBCF6',
        'green', 'tomato', 'darkred', 'black', 'cyan', 'lime'
    ])
    my_colors = np.array(['???????'] * coefs.shape[-1])
    i_col = 0
    new_grp_pts_x = []
    new_grp_pts_y = []
    new_grp_pts_col = []
    new_grp_pts_total = []

    for i_vertical, (params, acc, C) in enumerate(zip(
        coefs, accs, C_grid)):
        b_notset = my_colors == '???????'
        b_nonzeros = params == 0
        b_coefs_of_new_grp = np.logical_and(b_notset, b_nonzeros)
```

```
        #if i_vertical >= 17:
        #    import pdb; pdb.set_trace()

        if np.sum(b_coefs_of_new_grp) > 0:
            i_col += 1

            # we found a new subset that became 0
            for new_i in np.where(b_coefs_of_new_grp == True)[0]:
                # color all coefficients of the current group
                cur_col = my_palette[i_col]
                my_colors[new_i] = cur_col

            new_grp_pts_x.append(C)
            new_grp_pts_y.append(acc)
            new_grp_pts_col.append(cur_col)
            new_grp_pts_total.append(np.sum(b_nonzeros))

    if var_names is None:
        X_colnames = np.arange(n_feat) + 1
    else:
        X_colnames = var_names

    subplot_xlabel = '#nonzero coefficients'

    f, axarr = plt.subplots(nrows=n_rows, ncols=n_cols,
        figsize=(15, 10), facecolor='white')
    t, i_col = 0, 0

    for i_line in range(X.shape[-1]):
        axarr[i_col].plot(np.log10(C_grid)[::-1],
            coefs[:, i_line], label=X_colnames[i_line],
                color=my_colors[i_line], linewidth=1.5)

    # axarr[0].set_xticks(np.arange(len(C_grid)))
    # axarr[0].set_xticklabels(np.log10(C_grid))  #, rotation=75)
    axarr[i_col].set_xlabel(subplot_xlabel, fontsize=10)
    axarr[i_col].legend(loc='lower left', fontsize=11, markerscale=10,
fancybox=True, framealpha=0.5)
    axarr[0].grid(True)
    # axarr[i_col].set_ylabel('Item groups', fontsize=16)
    axarr[0].set_title('Penalized Logistic: Groups of selected variabl
es', fontsize=16)
    axarr[0].set_xticks(np.log10(C_grid)[::-1])
    axarr[0].set_xticklabels(nonzeros)

    # axarr[1].axis('off')
    #import pdb; pdb.set_trace()
    if unbiased_accs is not None:
        axarr[1].scatter(np.arange(len(unbiased_accs)), unbiased_accs,
color='orange',
```

```
                          linewidth=4, label='prediction accuracy (debiased
)', zorder=10)
    axarr[1].scatter(np.arange(len(accs)), accs, color='black',
                          linewidth=3, label='prediction accuracy', zorder=
10)
    # axarr[1].set_title('ACCURACY')
    axarr[1].set_ylim(0, 1.05)
    axarr[1].grid(True)
    # axarr[1].set_xticklabels(np.log10(C_grid), '')
    axarr[1].set_xticks(np.arange(n_verticals))
    axarr[1].set_xticklabels(nonzeros)
    axarr[1].set_xlabel(subplot_xlabel, fontsize=10)
    # axarr[1].set_ylabel('Out-of-sample accuracy', fontsize=16)
    axarr[1].legend(loc='lower left', fontsize=14, markerscale=1, fanc
ybox=True, framealpha=0.5)
    axarr[1].set_title('Penalized Logistic: Out-of-sample accuracy ($R
^2$ score)', fontsize=16)


def corrfunc(x, y, **kws):
    from scipy import stats
    r, _ = stats.pearsonr(x, y)
    ax = plt.gca()
    ax.annotate("r = {:.2f}".format(r),
                xy=(.1, .9), xycoords=ax.transAxes)
```

```
/Users/dengeman/anaconda3/lib/python3.5/site-packages/sklearn/cross_
validation.py:41: DeprecationWarning: This module was deprecated in
version 0.18 in favor of the model_selection module into which all t
he refactored classes and functions are moved. Also note that the in
terface of the new CV iterators are different from that of this modu
le. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
```

In [2]:
```
import statsmodels.api as sm

# https://github.com/statsmodels/statsmodels/issues/3931
from scipy import stats
stats.chisqprob = lambda chisq, df: stats.chi2.sf(chisq, df)

# https://datascience.stackexchange.com/questions/937/does-scikit-lear
n-have-forward-selection-stepwise-regression-algorithm
def fwd_stepwise_selection(X, y, initial_list=[], verbose=True):
    """ Perform a forward-backward feature selection
    based on p-value from statsmodels.api.OLS
    Arguments:
        X - pandas.DataFrame with candidate features
        y - list-like with the target
        initial_list - list of features to start with (column names of
X)
        threshold_in - include a feature if its p-value < threshold_in
        threshold_out - exclude a feature if its p-value > threshold_o
ut
        verbose - whether to print the sequence of inclusions and excl
usions
    Returns: list of selected features
    """
    included = list(initial_list)
    while len(included) < X.shape[1]:
        # forward step
        excluded = list(set(X.columns)-set(included))
        new_pval = pd.Series(index=excluded)
        for new_column in excluded:
            model = Logit(y, sm.add_constant(pd.DataFrame(X[included +
[new_column]]))).fit(disp=0)
            new_pval[new_column] = model.pvalues[new_column]
        best_pval = new_pval.min()
        best_feature = new_pval.argmin()
        included.append(best_feature)
        if verbose:
            print('Add  {:30} with p-value {:.6}'.format(best_feature,
best_pval))
    return included
```

In [3]:
```
# statistical helper functions
def compute_Logistic_regpath(X, y, C_grid):
    coef_list2 = []
    acc_list2 = []
    acc_unbiased_list2 = []
    nonzero_list2 = []
    for i_step, my_C in enumerate(C_grid):
        sample_accs = []
        sample_accs_unbiased = []
```

```
        sample_coef = []
        for i_subsample in range(100):
            folder = ShuffleSplit(n=len(y), n_iter=100, test_size=0.1,
                                  random_state=i_subsample)
            train_inds, test_inds = next(iter(folder))

            clf = LogisticRegression(C=my_C, random_state=i_subsample,
penalty='l1')

            clf.fit(X[train_inds, :], y[train_inds])

            # compute out-of-sample prediction accuracy

            acc = average_precision_score(
                y_true=y[test_inds],
                y_score=clf.predict(X[test_inds]),
                average='weighted')

            # get out-of-sample accuracy from unbiased linear model wi
th selected inputs
            b_vars_to_keep = np.squeeze(clf.coef_) != 0
            if np.sum(b_vars_to_keep) > 0:
                unbiased_lr = LogisticRegression(C=100000, random_stat
e=i_subsample, penalty='l2')
                unbiased_lr.fit(
                  X[train_inds, :][:, b_vars_to_keep], y[train_inds])

                unbiased_acc = average_precision_score(
                    y_true=y[test_inds],
                    y_score=unbiased_lr.predict(X[test_inds][:, b_vars
_to_keep]),
                    average='weighted')
            else:
                unbiased_acc = 0

            sample_accs.append(acc)
            sample_accs_unbiased.append(unbiased_acc)
            sample_coef.append(np.squeeze(clf.coef_))

        mean_coefs = np.mean(np.array(sample_coef), axis=0)
        coef_list2.append(mean_coefs)
        acc_at_step = np.mean(sample_accs)
        acc_list2.append(acc_at_step)
        acc_unbiased_list2.append(np.mean(sample_accs_unbiased))
        notzero = np.count_nonzero(mean_coefs)
        print("C: %.4f acc: %.2f active_coefs: %i" % (my_C, acc_at_ste
p, notzero))
        nonzero_list2.append(notzero)
    return np.array(coef_list2), np.array(acc_list2), np.array(nonzero
_list2), np.array(acc_unbiased_list2)
```

Heart dataset (ISL)

Dataset summary: These data contain a binary outcome HD for 303 patients who presented with chest pain (binary outcome).

In [4]:

```
import pandas as pd
df_heart = pd.read_csv('dataset_heart_ISL.csv').fillna(value=0)
feat_names = ['Age', u'Sex', u'RestBP', u'Chol', u'Fbs',
       u'RestECG', u'MaxHR', u'ExAng', u'Oldpeak', u'Slope', u'Ca', u'
Thal', u'ChestPain']
y = np.asarray(df_heart['AHD'] == 'Yes', dtype=np.int)

df_part1 = pd.DataFrame(StandardScaler().fit_transform(df_heart[feat_n
ames[:-2]].values), columns=feat_names[:-2])
df_part2 = pd.get_dummies(df_heart[feat_names[-2:]])
#pd.concat([df_part1, df_part2], axis=1)
X = np.hstack((df_part1.values, df_part2.values))
feat_names = list(df_part1.columns) + list(df_part2.columns)

cl2_prop = np.sum(y) * 100 / len(y)
print('Balance between class 1 : 2 is %.0f%% : %.0f%%' % (100 - cl2_pr
op, cl2_prop))
```

```
Balance between class 1 : 2 is 54% : 46%
```

In [5]:

```
df_heart
```

Out[5]:

	Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng
0	1	63	1	typical	145	233	1	2	150	0
1	2	67	1	asymptomatic	160	286	0	2	108	1
2	3	67	1	asymptomatic	120	229	0	2	129	1
3	4	37	1	nonanginal	130	250	0	0	187	0
4	5	41	0	nontypical	130	204	0	2	172	0
5	6	56	1	nontypical	120	236	0	0	178	0
6	7	62	0	asymptomatic	140	268	0	2	160	0
7	8	57	0	asymptomatic	120	354	0	0	163	1

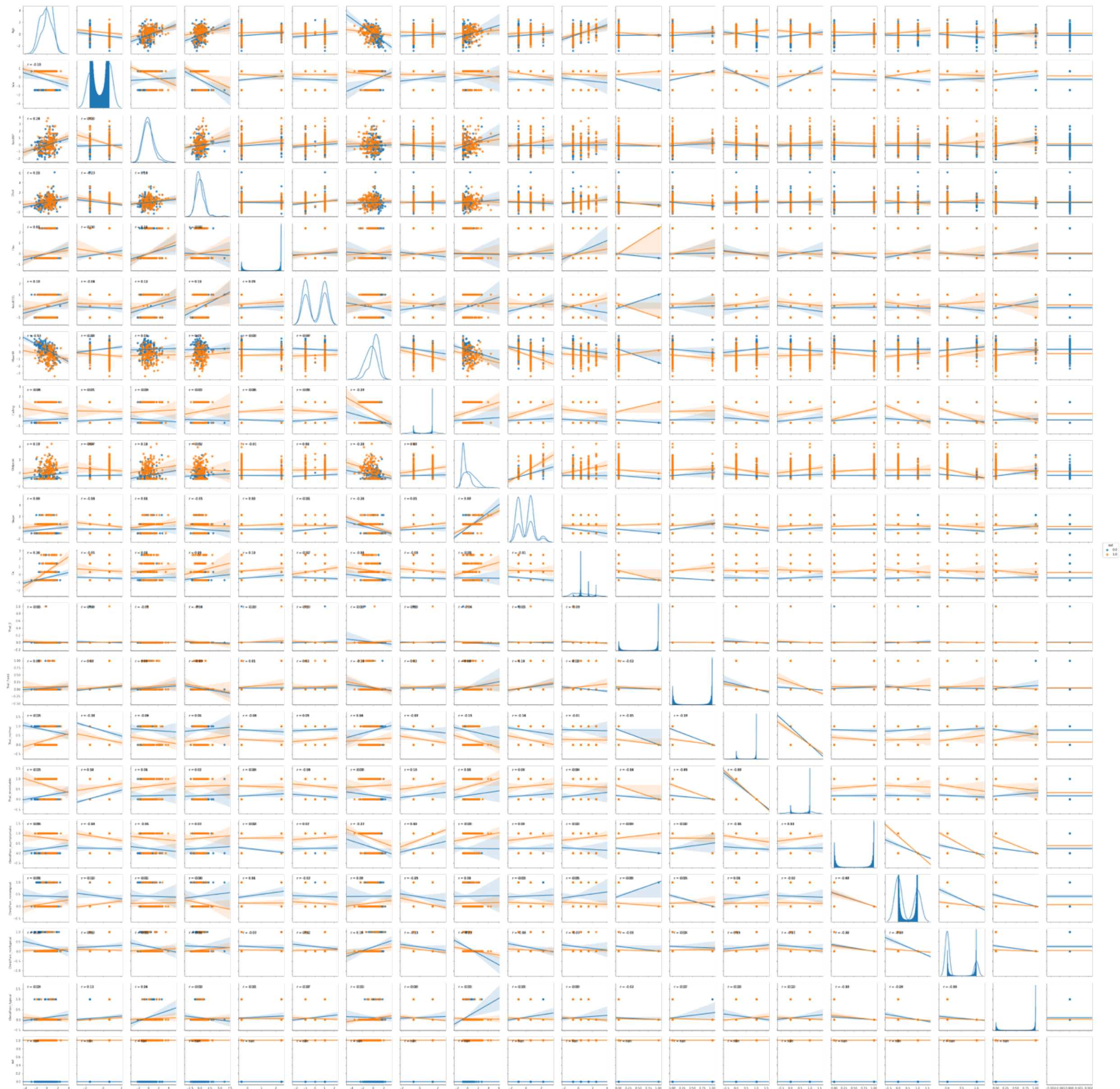
8	9	63	1	asymptomatic	130	254	0	2	147	0
9	10	53	1	asymptomatic	140	203	1	2	155	1
10	11	57	1	asymptomatic	140	192	0	0	148	0
11	12	56	0	nontypical	140	294	0	2	153	0
12	13	56	1	nonanginal	130	256	1	2	142	1
13	14	44	1	nontypical	120	263	0	0	173	0
14	15	52	1	nonanginal	172	199	1	0	162	0
15	16	57	1	nonanginal	150	168	0	0	174	0
16	17	48	1	nontypical	110	229	0	0	168	0
17	18	54	1	asymptomatic	140	239	0	0	160	0
18	19	48	0	nonanginal	130	275	0	0	139	0
19	20	49	1	nontypical	130	266	0	0	171	0
20	21	64	1	typical	110	211	0	2	144	1
21	22	58	0	typical	150	283	1	2	162	0
22	23	58	1	nontypical	120	284	0	2	160	0
23	24	58	1	nonanginal	132	224	0	2	173	0
24	25	60	1	asymptomatic	130	206	0	2	132	1
25	26	50	0	nonanginal	120	219	0	0	158	0
26	27	58	0	nonanginal	120	340	0	0	172	0
27	28	66	0	typical	150	226	0	0	114	0
28	29	43	1	asymptomatic	150	247	0	0	171	0
29	30	40	1	asymptomatic	110	167	0	2	114	1
...	...	...	...	...	...	...	...	...	...	...
273	274	71	0	asymptomatic	112	149	0	0	125	0
274	275	59	1	typical	134	204	0	0	162	0
275	276	64	1	typical	170	227	0	2	155	0
276	277	66	0	nonanginal	146	278	0	2	152	0
277	278	39	0	nonanginal	138	220	0	0	152	0
278	279	57	1	nontypical	154	232	0	2	164	0

279	280	58	0	asymptomatic	130	197	0	0	131	0
280	281	57	1	asymptomatic	110	335	0	0	143	1
281	282	47	1	nonanginal	130	253	0	0	179	0
282	283	55	0	asymptomatic	128	205	0	1	130	1
283	284	35	1	nontypical	122	192	0	0	174	0
284	285	61	1	asymptomatic	148	203	0	0	161	0
285	286	58	1	asymptomatic	114	318	0	1	140	0
286	287	58	0	asymptomatic	170	225	1	2	146	1
287	288	58	1	nontypical	125	220	0	0	144	0
288	289	56	1	nontypical	130	221	0	2	163	0
289	290	56	1	nontypical	120	240	0	0	169	0
290	291	67	1	nonanginal	152	212	0	2	150	0
291	292	55	0	nontypical	132	342	0	0	166	0
292	293	44	1	asymptomatic	120	169	0	0	144	1
293	294	63	1	asymptomatic	140	187	0	2	144	1
294	295	63	0	asymptomatic	124	197	0	0	136	1
295	296	41	1	nontypical	120	157	0	0	182	0
296	297	59	1	asymptomatic	164	176	1	2	90	0
297	298	57	0	asymptomatic	140	241	0	0	123	1
298	299	45	1	typical	110	264	0	0	132	0
299	300	68	1	asymptomatic	144	193	1	0	141	0
300	301	57	1	asymptomatic	130	131	0	0	115	1
301	302	57	0	nontypical	130	236	0	2	174	0
302	303	38	1	nonanginal	138	175	0	0	173	0

303 rows × 15 columns

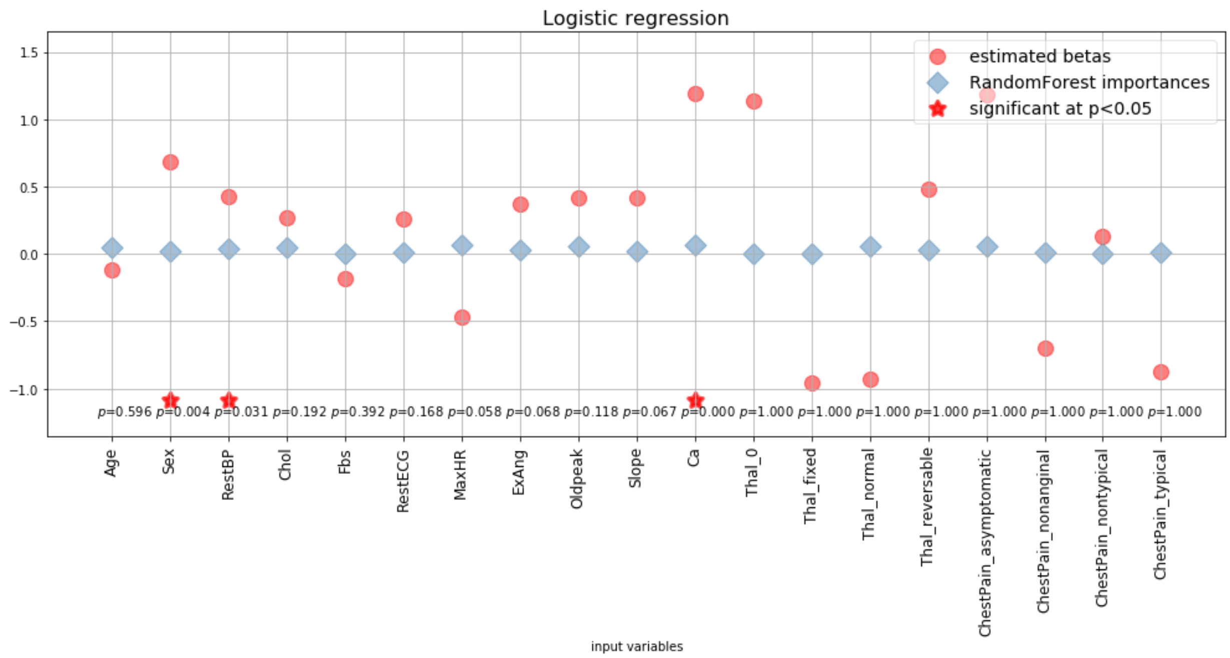
In [6]:
```
g = sns.pairplot(pd.DataFrame(np.hstack((X, y[:, None])), columns=feat
_names + ['out']),
                 kind="reg", diag_kind="kde", hue='out')
g.map_lower(corrfunc)
```

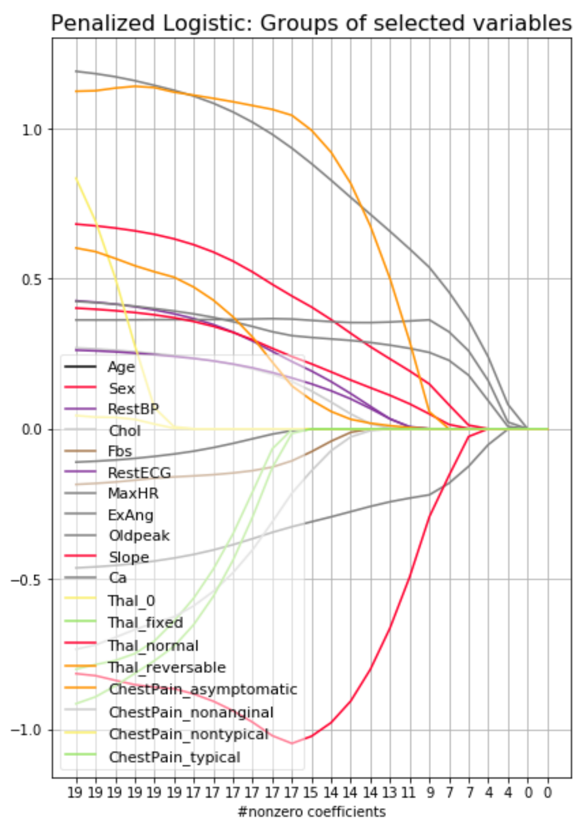
Out[6]: <seaborn.axisgrid.PairGrid at 0x10877af98>

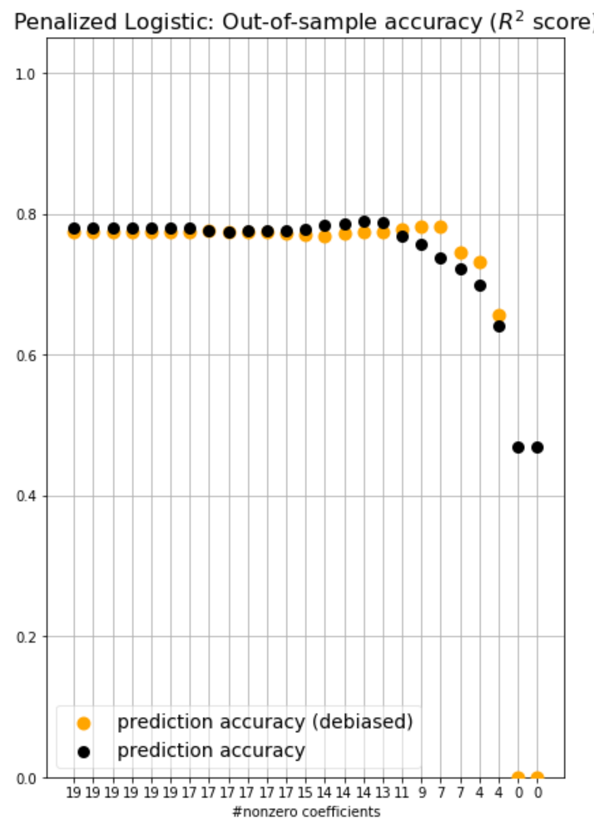


In [7]:
```
# ordinary linear model with logit loss
model = Logit(y, X)
res = model.fit(disp=0)
lr_coefs = res.params
lr_pvalues = res.pvalues

rf_cmp.fit(X, y)
rf_cmp.feature_importances_

# compute regularization paths of L1-penalized linear model with logit
loss
C_grid = np.logspace(+1, -2, 25)
coef_list, acc_list, nonzero_list, unbiased_acc_list = compute_Logisti
c_regpath(X, y, C_grid)

plot_lr(None, lr_coefs, lr_pvalues, feat_names, rf_cmp_coef=rf_cmp.fea
ture_importances_ * np.mean(np.abs(lr_coefs)))
plot_regr_paths(coef_list, acc_list, nonzero_list, C_grid, feat_names,
unbiased_acc_list)
```

```
C: 10.0000 acc: 0.78 active_coefs: 19
C: 7.4989 acc: 0.78 active_coefs: 19
C: 5.6234 acc: 0.78 active_coefs: 19
C: 4.2170 acc: 0.78 active_coefs: 19
C: 3.1623 acc: 0.78 active_coefs: 19
C: 2.3714 acc: 0.78 active_coefs: 19
C: 1.7783 acc: 0.78 active_coefs: 17
C: 1.3335 acc: 0.78 active_coefs: 17
C: 1.0000 acc: 0.78 active_coefs: 17
C: 0.7499 acc: 0.78 active_coefs: 17
C: 0.5623 acc: 0.78 active_coefs: 17
C: 0.4217 acc: 0.78 active_coefs: 17
C: 0.3162 acc: 0.78 active_coefs: 15
C: 0.2371 acc: 0.78 active_coefs: 14
C: 0.1778 acc: 0.79 active_coefs: 14
C: 0.1334 acc: 0.79 active_coefs: 14
C: 0.1000 acc: 0.79 active_coefs: 13
C: 0.0750 acc: 0.77 active_coefs: 11
C: 0.0562 acc: 0.76 active_coefs: 9
C: 0.0422 acc: 0.74 active_coefs: 7
C: 0.0316 acc: 0.72 active_coefs: 7
C: 0.0237 acc: 0.70 active_coefs: 4
C: 0.0178 acc: 0.64 active_coefs: 4
C: 0.0133 acc: 0.47 active_coefs: 0
C: 0.0100 acc: 0.47 active_coefs: 0
```


Logistic regression
estimated betas
RandomForest importances
significant at p<0.05
p=0.596 p=0.004 p=0.031 p=0.192 p=0.392 p=0.168 p=0.058 p=0.068 p=0.118 p=0.067 p=0.000 p=1.000 p=1.000 p=1.000 p=1.000 p=1.000 p=1.000 p=1.000 p=1.000
Age
Sex
RestBP
Chol
Fbs
RestECG
MaxHR
ExAng
Oldpeak
Slope
Ca
Thal_0
Thal_fixed
Thal_normal
Thal_reversable
ChestPain_asymptomatic
ChestPain_nonanginal
ChestPain_nontypical
ChestPain_typical
input variables


Penalized Logistic: Groups of selected variables
Age
Sex
RestBP
Chol
Fbs
RestECG
MaxHR
ExAng
Oldpeak
Slope
Ca
Thal_0
Thal_fixed
Thal_normal
Thal_reversable
ChestPain_asymptomatic
ChestPain_nonanginal
ChestPain_nontypical
ChestPain_typical
19 19 19 19 19 19 17 17 17 17 17 17 15 14 14 14 13 11 9 7 7 4 4 0 0
#nonzero coefficients


Penalized Logistic: Out-of-sample accuracy (R^2 score)
prediction accuracy (debiased)
prediction accuracy
19 19 19 19 19 19 17 17 17 17 17 17 15 14 14 14 13 11 9 7 7 4 4 0 0
#nonzero coefficients

In [8]:
```
sel_w_pvals = fwd_stepwise_selection(pd.DataFrame(X, columns=feat_name
s), y, verbose=True)
print('Forward-stepwise selection: ' + ' -> '.join(sel_w_pvals))
```

```
Add  Thal_normal                    with p-value 1.01554e-17
Add  ChestPain_asymptomatic         with p-value 8.76513e-12
Add  Ca                             with p-value 5.44918e-08
Add  Oldpeak                        with p-value 0.000292995
Add  ExAng                          with p-value 0.0177969
Add  RestECG                        with p-value 0.0487469
Add  Sex                            with p-value 0.0720185
Add  MaxHR                          with p-value 0.0784324
Add  RestBP                         with p-value 0.0625983
Add  Thal_fixed                     with p-value 0.062896
Add  Slope                          with p-value 0.116818
Add  ChestPain_nontypical           with p-value 0.0956932
Add  Chol                           with p-value 0.225965
Add  Fbs                            with p-value 0.414937
Add  Age                            with p-value 0.585244
Add  Thal_0                         with p-value 0.735713
Add  ChestPain_nonanginal           with p-value 0.782808
Add  ChestPain_typical              with p-value 1.0
Add  Thal_reversable                with p-value 1.0
Forward-stepwise selection: Thal_normal -> ChestPain_asymptomatic ->
Ca -> Oldpeak -> ExAng -> RestECG -> Sex -> MaxHR -> RestBP -> Thal_
fixed -> Slope -> ChestPain_nontypical -> Chol -> Fbs -> Age -> Thal
_0 -> ChestPain_nonanginal -> ChestPain_typical -> Thal_reversable
```

In [9]:
```
res.summary(xname=feat_names)
```

Out[9]:

Logit Regression Results

Dep. Variable:	y	**No. Observations:**	303
Model:	Logit	**Df Residuals:**	285
Method:	MLE	**Df Model:**	17
Date:	Mon, 21 May 2018	**Pseudo R-squ.:**	0.5285
Time:	00:00:16	**Log-Likelihood:**	-98.548
converged:	True	**LL-Null:**	-208.99
		LLR p-value:	1.745e-37

	coef	std err	z	P>|z|	[0.025	0.975]
Age	-0.1180	0.223	-0.530	0.596	-0.554	0.318

Sex	0.6847	0.239	2.859	0.004	0.215	1.154
RestBP	0.4237	0.196	2.161	0.031	0.039	0.808
Chol	0.2679	0.205	1.305	0.192	-0.134	0.670
Fbs	-0.1784	0.208	-0.856	0.392	-0.587	0.230
RestECG	0.2583	0.187	1.379	0.168	-0.109	0.625
MaxHR	-0.4647	0.245	-1.895	0.058	-0.946	0.016
ExAng	0.3708	0.203	1.823	0.068	-0.028	0.769
Oldpeak	0.4189	0.268	1.564	0.118	-0.106	0.944
Slope	0.4164	0.228	1.830	0.067	-0.030	0.862
Ca	1.1919	0.254	4.692	0.000	0.694	1.690
Thal_0	1.1389	1.06e+07	1.08e-07	1.000	-2.07e+07	2.07e+07
Thal_fixed	-0.9576	1.06e+07	-9.05e-08	1.000	-2.07e+07	2.07e+07
Thal_normal	-0.9272	1.06e+07	-8.76e-08	1.000	-2.07e+07	2.07e+07
Thal_reversable	0.4805	1.06e+07	4.54e-08	1.000	-2.07e+07	2.07e+07
ChestPain_asymptomatic	1.1782	1.06e+07	1.11e-07	1.000	-2.07e+07	2.07e+07
ChestPain_nonanginal	-0.6998	1.06e+07	-6.62e-08	1.000	-2.07e+07	2.07e+07
ChestPain_nontypical	0.1346	1.06e+07	1.27e-08	1.000	-2.07e+07	2.07e+07
ChestPain_typical	-0.8784	1.06e+07	-8.3e-08	1.000	-2.07e+07	2.07e+07

conclusion: only 1 of 3 significant variables is among the 4 most predictive ones

South African Heart dataset: many significant but one most predictive

Dataset summary (ESL): A retrospective sample of males in a heart-disease high-risk region of the Western Cape, South Africa. There are roughly two controls per case of coronary heart disease. Many of the coronary heart disease positive men have undergone blood pressure reduction treatment and other programs to reduce their risk factors after their coronary heart disease event. In some cases the measurements were made after these treatments. These data are taken from a larger dataset, described in Rousseauw et al, 1983, South African Medical Journal.

Based on this data, does having a family history of coronary heart disease affect a patients chance of having coronary heart disease? Does this result change for patients younger than 40 years old? What about for patients aged 40 years or older?

sbp systolic blood pressure tobacco cumulative tobacco (kg) ldl low densiity lipoprotein cholesterol adiposity famhist family history of heart disease (Present, Absent) typea type-A behavior obesity alcohol current alcohol consumption age age at onset chd response, coronary heart disease

In [10]:
```
import pandas as pd
df_africa = pd.read_excel('dataset_south_african_heart_disease.xls')

df_africa
```

Out[10]:

	row	sbp	tobacco	ldl	adiposity	famhist	typea	obesity	alcohol	age	chd
0	1	160	12.00	5.73	23.11	Present	49	25.30	97.20	52	1
1	2	144	0.01	4.41	28.61	Absent	55	28.87	2.06	63	1
2	3	118	0.08	3.48	32.28	Present	52	29.14	3.81	46	0
3	4	170	7.50	6.41	38.03	Present	51	31.99	24.26	58	1
4	5	134	13.60	3.50	27.78	Present	60	25.99	57.34	49	1
5	6	132	6.20	6.47	36.21	Present	62	30.77	14.14	45	0
6	7	142	4.05	3.38	16.20	Absent	59	20.81	2.62	38	0
7	8	114	4.08	4.59	14.60	Present	62	23.11	6.72	58	1
8	9	114	0.00	3.83	19.40	Present	49	24.86	2.49	29	0
9	10	132	0.00	5.80	30.96	Present	69	30.11	0.00	53	1
10	11	206	6.00	2.95	32.27	Absent	72	26.81	56.06	60	1
11	12	134	14.10	4.44	22.39	Present	65	23.09	0.00	40	1
12	13	118	0.00	1.88	10.05	Absent	59	21.57	0.00	17	0
13	14	132	0.00	1.87	17.21	Absent	49	23.63	0.97	15	0

14	15	112	9.65	2.29	17.20	Present	54	23.53	0.68	53	0
15	16	117	1.53	2.44	28.95	Present	35	25.89	30.03	46	0
16	17	120	7.50	15.33	22.00	Absent	60	25.31	34.49	49	0
17	18	146	10.50	8.29	35.36	Present	78	32.73	13.89	53	1
18	19	158	2.60	7.46	34.07	Present	61	29.30	53.28	62	1
19	20	124	14.00	6.23	35.96	Present	45	30.09	0.00	59	1
20	21	106	1.61	1.74	12.32	Absent	74	20.92	13.37	20	1
21	22	132	7.90	2.85	26.50	Present	51	26.16	25.71	44	0
22	23	150	0.30	6.38	33.99	Present	62	24.64	0.00	50	0
23	24	138	0.60	3.81	28.66	Absent	54	28.70	1.46	58	0
24	25	142	18.20	4.34	24.38	Absent	61	26.19	0.00	50	0
25	26	124	4.00	12.42	31.29	Present	54	23.23	2.06	42	1
26	27	118	6.00	9.65	33.91	Absent	60	38.80	0.00	48	0
27	28	145	9.10	5.24	27.55	Absent	59	20.96	21.60	61	1
28	29	144	4.09	5.55	31.40	Present	60	29.43	5.55	56	0
29	30	146	0.00	6.62	25.69	Absent	60	28.07	8.23	63	1
...	...	...	...	...	...	...	...	...	...	...	...
432	434	136	0.00	4.00	19.06	Absent	40	21.94	2.06	16	0
433	435	120	0.00	2.46	13.39	Absent	47	22.01	0.51	18	0
434	436	132	0.00	3.55	8.66	Present	61	18.50	3.87	16	0
435	437	136	0.00	1.77	20.37	Absent	45	21.51	2.06	16	0
436	438	138	0.00	1.86	18.35	Present	59	25.38	6.51	17	0
437	439	138	0.06	4.15	20.66	Absent	49	22.59	2.49	16	0
438	440	130	1.22	3.30	13.65	Absent	50	21.40	3.81	31	0
439	441	130	4.00	2.40	17.42	Absent	60	22.05	0.00	40	0
440	442	110	0.00	7.14	28.28	Absent	57	29.00	0.00	32	0
441	443	120	0.00	3.98	13.19	Present	47	21.89	0.00	16	0
442	444	166	6.00	8.80	37.89	Absent	39	28.70	43.20	52	0
443	445	134	0.57	4.75	23.07	Absent	67	26.33	0.00	37	0

444	446	142	3.00	3.69	25.10	Absent	60	30.08	38.88	27	0
445	447	136	2.80	2.53	9.28	Present	61	20.70	4.55	25	0
446	448	142	0.00	4.32	25.22	Absent	47	28.92	6.53	34	1
447	449	130	0.00	1.88	12.51	Present	52	20.28	0.00	17	0
448	450	124	1.80	3.74	16.64	Present	42	22.26	10.49	20	0
449	451	144	4.00	5.03	25.78	Present	57	27.55	90.00	48	1
450	452	136	1.81	3.31	6.74	Absent	63	19.57	24.94	24	0
451	453	120	0.00	2.77	13.35	Absent	67	23.37	1.03	18	0
452	454	154	5.53	3.20	28.81	Present	61	26.15	42.79	42	0
453	455	124	1.60	7.22	39.68	Present	36	31.50	0.00	51	1
454	456	146	0.64	4.82	28.02	Absent	60	28.11	8.23	39	1
455	457	128	2.24	2.83	26.48	Absent	48	23.96	47.42	27	1
456	458	170	0.40	4.11	42.06	Present	56	33.10	2.06	57	0
457	459	214	0.40	5.98	31.72	Absent	64	28.45	0.00	58	0
458	460	182	4.20	4.41	32.10	Absent	52	28.61	18.72	52	1
459	461	108	3.00	1.59	15.23	Absent	40	20.09	26.64	55	0
460	462	118	5.40	11.61	30.79	Absent	64	27.35	23.97	40	0
461	463	132	0.00	4.82	33.41	Present	62	14.70	0.00	46	1

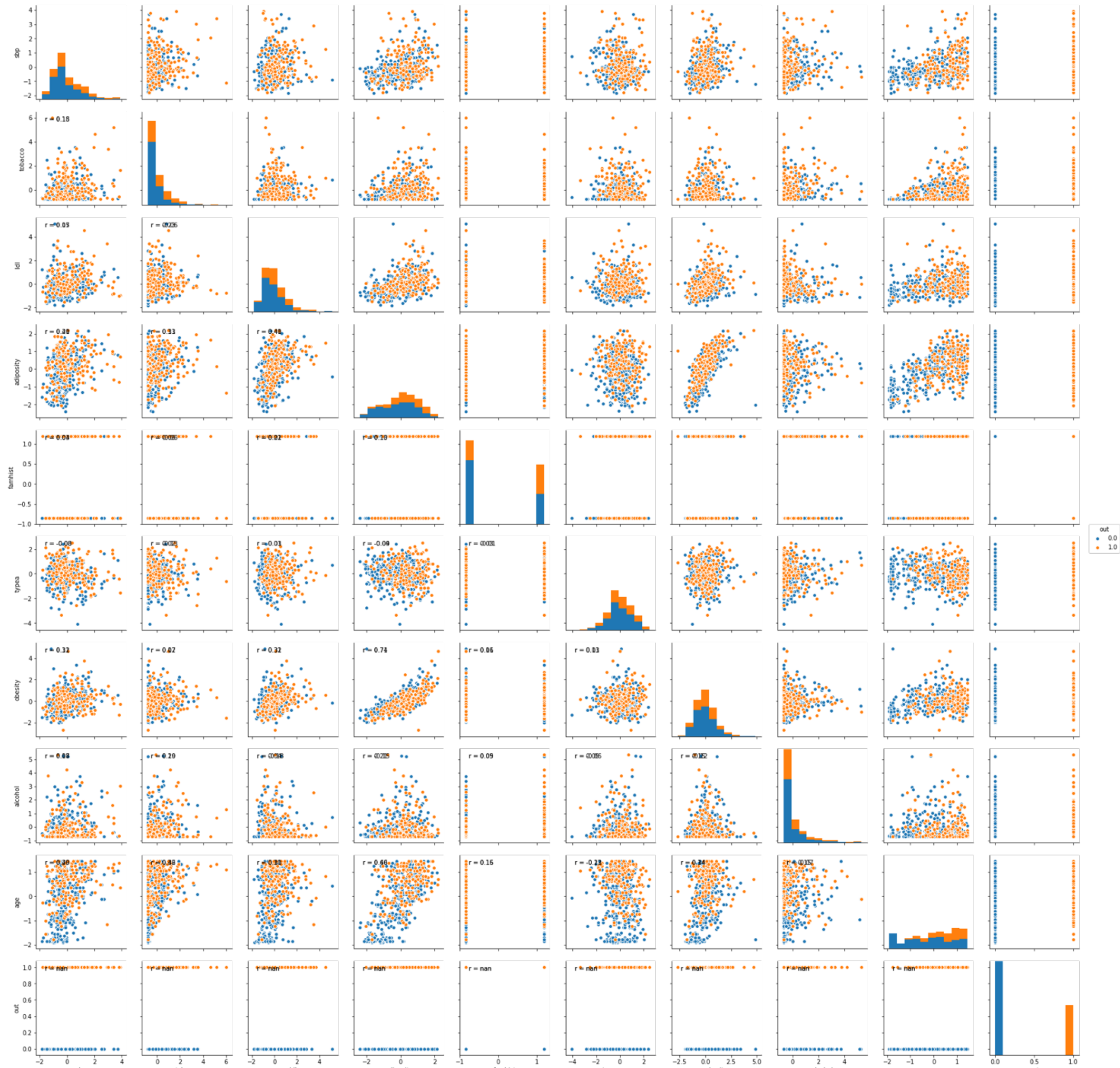
462 rows × 11 columns

In [11]:
```
feat_names = ['sbp', u'tobacco', u'ldl', u'adiposity', u'famhist', u't
ypea',
       u'obesity', u'alcohol', u'age']
df_africa['famhist'] = pd.get_dummies(df_africa['famhist'], drop_first
=True)
X = StandardScaler().fit_transform(df_africa[feat_names].values)
y = df_africa['chd'].values

cl2_prop = np.sum(y) * 100 / len(y)
print('Balance between class 1 : 2 is %.0f%% : %.0f%%' % (100 - cl2_pr
op, cl2_prop))
```

```
Balance between class 1 : 2 is 65% : 35%
```

In [12]:
```
g = sns.pairplot(pd.DataFrame(np.hstack((X, y[:, None])), columns=feat
_names + ['out']),
                 hue='out')
g.map_lower(corrfunc)
```

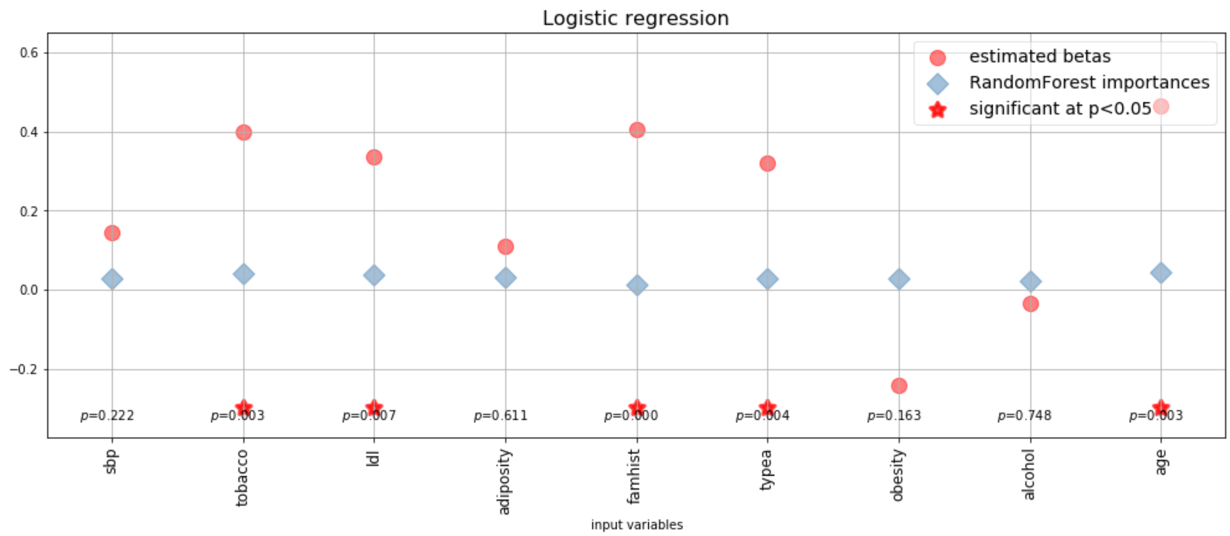
Out[12]: <seaborn.axisgrid.PairGrid at 0x135653198>

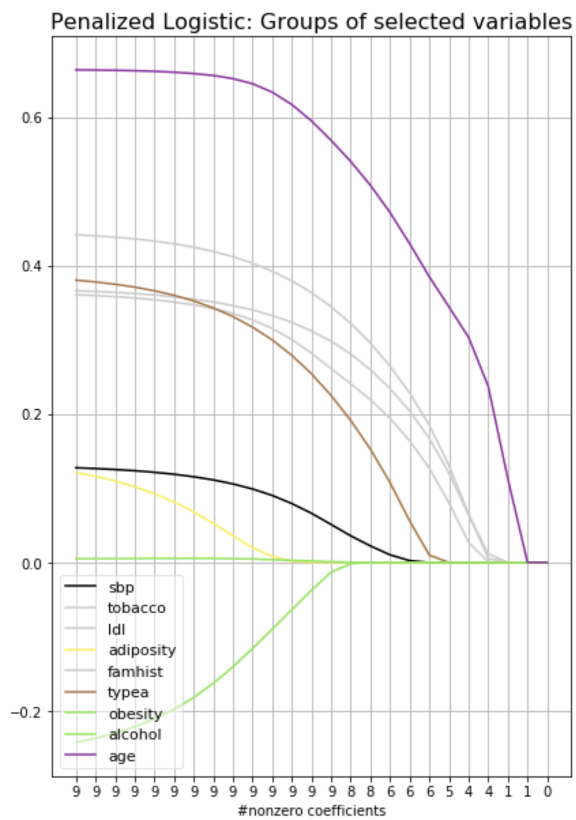


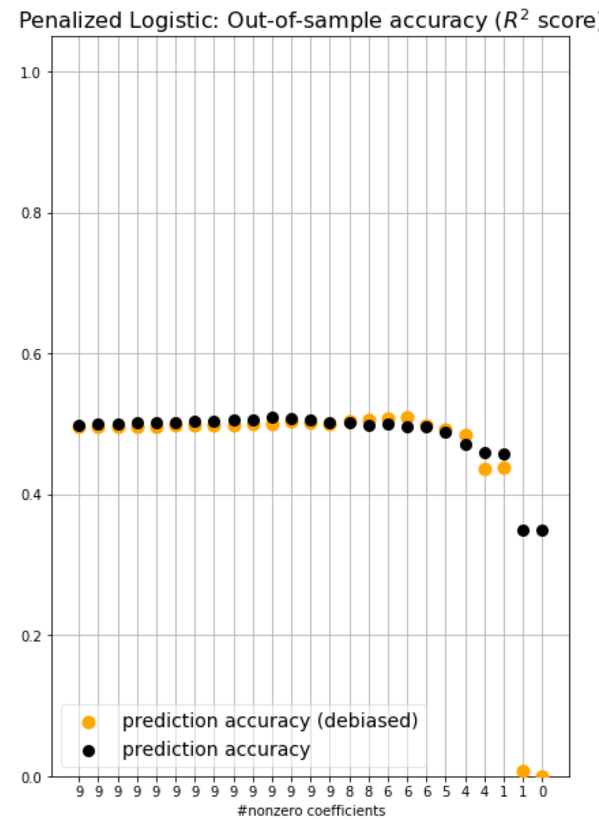
In [13]:

```
# ordinary linear model with logit loss
model = Logit(y, X)
res = model.fit(disp=0)
lr_coefs = res.params
lr_pvalues = res.pvalues

rf_cmp.fit(X, y)
rf_cmp.feature_importances_

# compute regularization paths of L1-penalized linear model with logit loss
C_grid = np.logspace(+.5, -2, 25)
coef_list, acc_list, nonzero_list, unbiased_acc_list = compute_Logistic_regpath(X, y, C_grid)

plot_lr(None, lr_coefs, lr_pvalues, feat_names, rf_cmp_coef=rf_cmp.feature_importances_ * np.mean(np.abs(lr_coefs)))
plot_regr_paths(coef_list, acc_list, nonzero_list, C_grid, feat_names, unbiased_acc_list)
```

```
C: 3.1623 acc: 0.50 active_coefs: 9
C: 2.4879 acc: 0.50 active_coefs: 9
C: 1.9573 acc: 0.50 active_coefs: 9
C: 1.5399 acc: 0.50 active_coefs: 9
C: 1.2115 acc: 0.50 active_coefs: 9
C: 0.9532 acc: 0.50 active_coefs: 9
C: 0.7499 acc: 0.50 active_coefs: 9
C: 0.5900 acc: 0.50 active_coefs: 9
C: 0.4642 acc: 0.51 active_coefs: 9
C: 0.3652 acc: 0.51 active_coefs: 9
C: 0.2873 acc: 0.51 active_coefs: 9
C: 0.2260 acc: 0.51 active_coefs: 9
C: 0.1778 acc: 0.51 active_coefs: 9
C: 0.1399 acc: 0.50 active_coefs: 9
C: 0.1101 acc: 0.50 active_coefs: 8
C: 0.0866 acc: 0.50 active_coefs: 8
C: 0.0681 acc: 0.50 active_coefs: 6
C: 0.0536 acc: 0.50 active_coefs: 6
C: 0.0422 acc: 0.50 active_coefs: 6
C: 0.0332 acc: 0.49 active_coefs: 5
C: 0.0261 acc: 0.47 active_coefs: 4
C: 0.0205 acc: 0.46 active_coefs: 4
C: 0.0162 acc: 0.46 active_coefs: 1
C: 0.0127 acc: 0.35 active_coefs: 1
C: 0.0100 acc: 0.35 active_coefs: 0
```


Logistic regression
estimated betas
RandomForest importances
significant at p<0.05
p=0.222
p=0.003
p=0.007
p=0.611
p=0.000
p=0.004
p=0.163
p=0.748
p=0.003
sbp
tobacco
ldl
adiposity
famhist
typea
obesity
alcohol
age
input variables


Penalized Logistic: Groups of selected variables
sbp
tobacco
ldl
adiposity
famhist
typea
obesity
alcohol
age
#nonzero coefficients


Penalized Logistic: Out-of-sample accuracy (R^2 score)
prediction accuracy (debiased)
prediction accuracy
#nonzero coefficients

In [14]:
```
sel_w_pvals = fwd_stepwise_selection(pd.DataFrame(X, columns=feat_name
s), y, verbose=True)
print('Forward-stepwise selection: ' +  ' -> '.join(sel_w_pvals))
```

```
Add  age                                  with p-value 5.75792e-14
Add  famhist                              with p-value 1.57789e-05
Add  typea                                with p-value 0.00121621
Add  tobacco                              with p-value 0.00145147
Add  ldl                                  with p-value 0.00320907
Add  obesity                              with p-value 0.196376
Add  sbp                                  with p-value 0.232859
Add  adiposity                            with p-value 0.524079
Add  alcohol                              with p-value 0.97835
Forward-stepwise selection: age -> famhist -> typea -> tobacco -> ld
l -> obesity -> sbp -> adiposity -> alcohol
```

conclusion:

- 5 significant variables, but only 1 of them achieve comparable prediction in new patients
- the most predictive feature is the health aspect that we can change least - age

In [15]:
```
res.summary(xname=feat_names)
```

Out[15]:

Logit Regression Results

Dep. Variable:	y	**No. Observations:**	462
Model:	Logit	**Df Residuals:**	453
Method:	MLE	**Df Model:**	8
Date:	Mon, 21 May 2018	**Pseudo R-squ.:**	0.1085
Time:	00:00:35	**Log-Likelihood:**	-265.73
converged:	True	**LL-Null:**	-298.05
		LLR p-value:	5.656e-11

	coef	std err	z	P>|z|	[0.025	0.975]
sbp	0.1428	0.117	1.220	0.222	-0.087	0.372
tobacco	0.3971	0.132	3.006	0.003	0.138	0.656
ldl	0.3351	0.124	2.700	0.007	0.092	0.578
adiposity	0.1093	0.215	0.509	0.611	-0.311	0.530
famhist	0.4047	0.107	3.795	0.000	0.196	0.614
typea	0.3198	0.111	2.885	0.004	0.103	0.537
obesity	-0.2427	0.174	-1.395	0.163	-0.584	0.098
alcohol	-0.0344	0.107	-0.321	0.748	-0.244	0.175
age	0.4647	0.158	2.942	0.003	0.155	0.774

Coronary Heart Disease: 4 significant, but 1 most predictive

Dataset summary: This dataset is from the Duke University Cardiovascular Disease Databank and consists of 3504 patients and 6 variables. The patients were referred to Duke University Medical Center for chest pain. Some interesting analyses include predicting the probability of significant (>= 75% diameter narrowing in at least one important coronary artery) coronary disease, and predicting the probability of severe coronary disease given that some significant disease is "ruled in." The first analysis would use sigdz as a response variable, and the second would use tvdlm on the subset of patients having sigdz=1. Severe coronary disease is defined as three-vessel or left main disease and is denoted by tvdlm=1. sex=0 for males, 1 for females.

3504 observations and 6 variables

In [16]:
```
import pandas as pd
df_coro = pd.read_excel('dataset_coronary_catheder.xls').dropna(how='a
ny')

df_coro
```

```
*** No CODEPAGE record, no encoding_override: will use 'ascii'
```

Out[16]:

	sex	age	cad.dur	choleste	sigdz	tvdlm
0	0	73	132	268.0	1	1.0
1	0	68	85	120.0	1	1.0
3	1	58	86	245.0	0	0.0
4	1	56	7	269.0	0	0.0
7	0	41	15	247.0	1	0.0
11	0	35	44	257.0	0	0.0
13	0	58	7	168.0	1	0.0
14	0	81	2	246.0	1	1.0
15	0	58	79	221.0	1	1.0
17	0	47	6	272.0	1	0.0
18	0	66	8	257.0	1	0.0
19	0	48	69	236.0	1	1.0
20	1	52	30	240.0	0	0.0
21	0	67	48	274.0	1	1.0

23	1	57	30	261.0	0	0.0
24	0	53	25	273.0	1	1.0
27	0	62	87	255.0	1	0.0
29	0	48	22	187.0	1	1.0
30	0	49	12	252.0	1	0.0
31	1	59	3	200.0	1	0.0
33	1	58	1	246.0	1	1.0
35	1	53	120	250.0	0	0.0
36	0	55	213	241.0	1	1.0
37	1	57	122	346.0	1	1.0
38	0	69	1	184.0	1	1.0
39	0	65	100	195.0	1	1.0
43	0	54	3	195.0	1	0.0
44	0	53	60	278.0	1	1.0
45	0	37	12	190.0	1	0.0
46	1	63	180	263.0	0	0.0
...	...	...	...	...	...	...
3461	0	59	172	198.0	1	1.0
3462	1	54	36	239.0	0	0.0
3463	1	60	220	272.0	1	0.0
3465	0	53	7	270.0	0	0.0
3466	0	69	290	258.0	1	1.0
3467	1	53	85	220.0	0	0.0
3468	0	62	29	220.0	1	0.0
3470	1	63	86	280.0	0	0.0
3471	1	65	6	197.0	1	1.0
3474	0	68	176	228.0	1	1.0
3477	0	54	18	185.0	0	0.0
3478	0	72	234	274.0	1	1.0

3479	0	65	156	230.0	1	0.0
3480	1	65	14	29.0	1	1.0
3481	0	64	0	161.0	1	0.0
3482	0	67	96	208.0	1	1.0
3483	1	38	2	255.0	0	0.0
3484	1	71	69	226.0	1	0.0
3485	0	55	11	182.0	1	1.0
3486	0	60	8	200.0	1	1.0
3487	1	58	119	165.0	1	1.0
3489	0	56	26	233.0	1	0.0
3490	0	56	152	208.0	1	1.0
3491	0	60	1	142.0	1	0.0
3493	1	53	24	225.0	1	1.0
3494	1	60	6	195.0	1	0.0
3496	1	42	32	164.0	0	0.0
3497	1	46	45	170.0	0	0.0
3498	0	46	1	196.0	1	0.0
3499	0	58	14	295.0	1	0.0

2258 rows × 6 columns

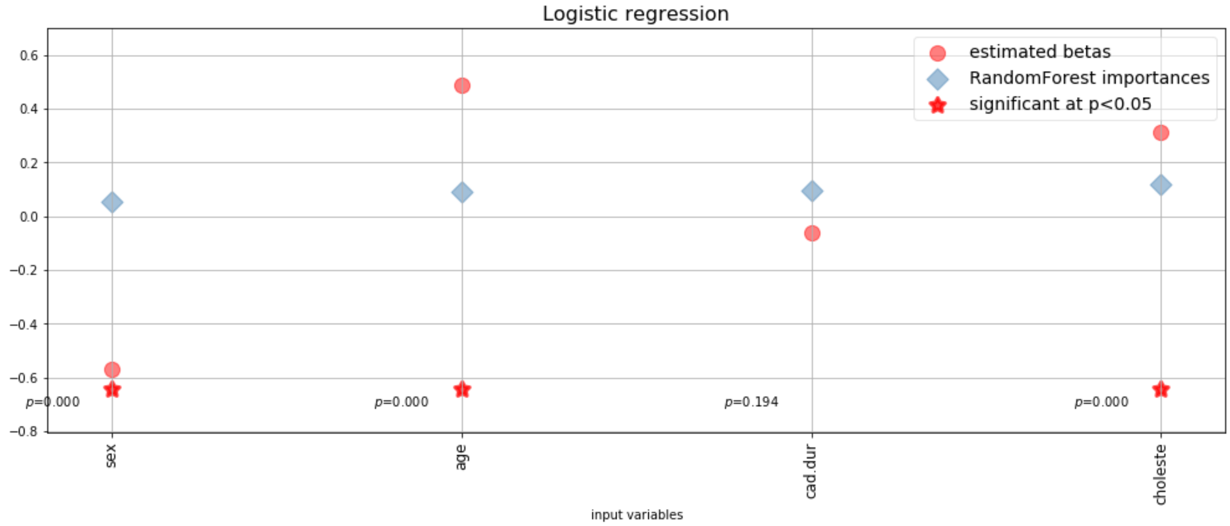
In [17]:

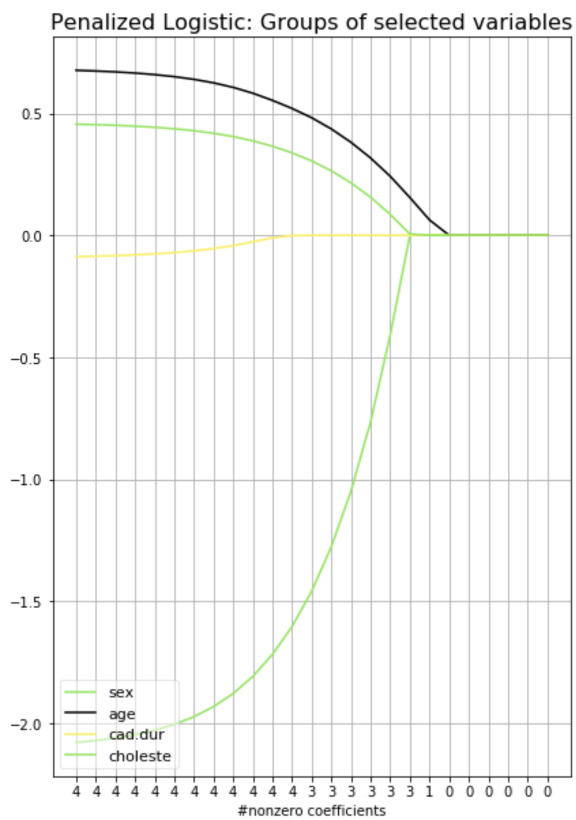
```
feat_names = ['sex', 'age', 'cad.dur', 'choleste']
feat_continuous = ['age', 'cad.dur', 'choleste']
df_coro[feat_continuous] = StandardScaler().fit_transform(df_coro[feat
_continuous])
#y = df_coro['tvdlm'].values
y = df_coro['sigdz'].values
df_coro = df_coro[feat_names]
X = df_coro.values

cl2_prop = np.sum(y) * 100 / len(y)
print('Balance between class 1 : 2 is %.0f%% : %.0f%%' % (100 - cl2_pr
op, cl2_prop))
```

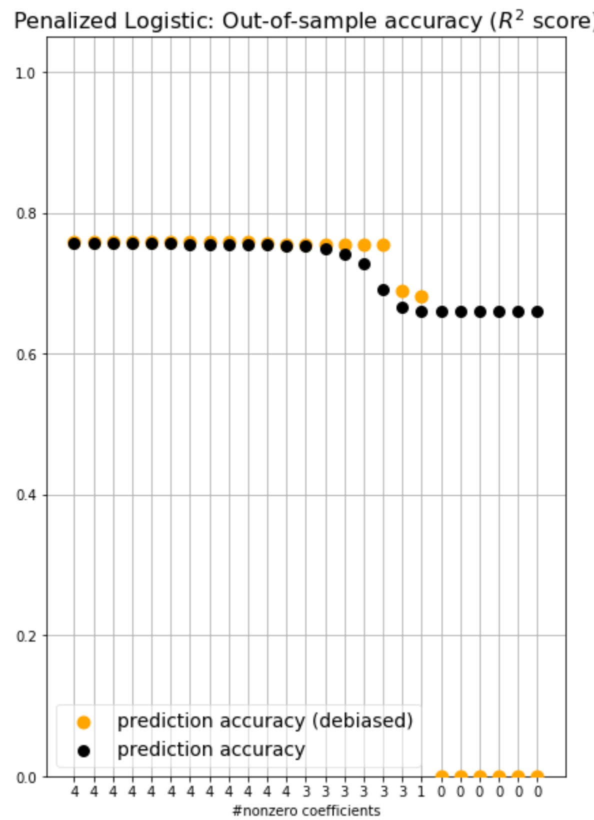
```
Balance between class 1 : 2 is 34% : 66%
```

In [18]:
```
# ordinary linear model with logit loss
model = Logit(y, X)
res = model.fit(disp=0)
lr_coefs = res.params
lr_pvalues = res.pvalues

rf_cmp.fit(X, y)
rf_cmp.feature_importances_

# compute regularization paths of L1-penalized linear model with logit
loss
C_grid = np.logspace(+.01, -3, 25)
coef_list, acc_list, nonzero_list, unbiased_acc_list = compute_Logisti
c_regpath(X, y, C_grid)

plot_lr(None, lr_coefs, lr_pvalues, feat_names, rf_cmp_coef=rf_cmp.fea
ture_importances_ * np.mean(np.abs(lr_coefs)))
plot_regr_paths(coef_list, acc_list, nonzero_list, C_grid, feat_names,
unbiased_acc_list)
```

```
C: 1.0233 acc: 0.76 active_coefs: 4
C: 0.7666 acc: 0.76 active_coefs: 4
C: 0.5743 acc: 0.76 active_coefs: 4
C: 0.4303 acc: 0.76 active_coefs: 4
C: 0.3224 acc: 0.76 active_coefs: 4
C: 0.2415 acc: 0.76 active_coefs: 4
C: 0.1809 acc: 0.76 active_coefs: 4
C: 0.1355 acc: 0.76 active_coefs: 4
C: 0.1015 acc: 0.76 active_coefs: 4
C: 0.0761 acc: 0.75 active_coefs: 4
C: 0.0570 acc: 0.75 active_coefs: 4
C: 0.0427 acc: 0.75 active_coefs: 4
C: 0.0320 acc: 0.75 active_coefs: 3
C: 0.0240 acc: 0.75 active_coefs: 3
C: 0.0180 acc: 0.74 active_coefs: 3
C: 0.0135 acc: 0.73 active_coefs: 3
C: 0.0101 acc: 0.69 active_coefs: 3
C: 0.0075 acc: 0.67 active_coefs: 3
C: 0.0057 acc: 0.66 active_coefs: 1
C: 0.0042 acc: 0.66 active_coefs: 0
C: 0.0032 acc: 0.66 active_coefs: 0
C: 0.0024 acc: 0.66 active_coefs: 0
C: 0.0018 acc: 0.66 active_coefs: 0
C: 0.0013 acc: 0.66 active_coefs: 0
C: 0.0010 acc: 0.66 active_coefs: 0
```







In [19]:

```
sel_w_pvals = fwd_stepwise_selection(pd.DataFrame(X, columns=feat_name
s), y, verbose=True)
print('Forward-stepwise selection: ' +  ' -> '.join(sel_w_pvals))
```

```
Add  sex                              with p-value 6.06864e-65
Add  age                              with p-value 8.65825e-34
Add  choleste                         with p-value 5.9199e-17
Add  cad.dur                          with p-value 0.0944686
Forward-stepwise selection: sex -> age -> choleste -> cad.dur
```

In [20]:
```
res.summary(xname=feat_names)
```

Out[20]:

Logit Regression Results

Dep. Variable:	y	**No. Observations:**	2258
Model:	Logit	**Df Residuals:**	2254
Method:	MLE	**Df Model:**	3
Date:	Mon, 21 May 2018	**Pseudo R-squ.:**	-0.01608
Time:	00:00:50	**Log-Likelihood:**	-1470.9
converged:	True	**LL-Null:**	-1447.6
		LLR p-value:	1.000

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
sex	-0.5702	0.083	-6.903	0.000	-0.732	-0.408
age	0.4863	0.049	9.969	0.000	0.391	0.582
cad.dur	-0.0598	0.046	-1.300	0.194	-0.150	0.030
choleste	0.3118	0.045	6.868	0.000	0.223	0.401

Excercise

Compare infrerence with prediction using the infpred plot from the notebook on regression problems.