{"cells":[{"metadata":{"_cell_guid":"866aea52-2435-49ea-914b-f97731e6a2bc","_uuid":"9346173c3f37fc70ee901cb8d505480361899da4","collapsed":true,"trusted":true},"cell_type":"code","source":"","execution_count":null,"outputs":[]},{"metadata":{"_cell_guid":"90b22ace-528d-477d-8880-6d9a1c979d7e","_uuid":"a25af58780d7ef5c499211f6dccfda2011b689a0"},"cell_type":"markdown","source":"# Churn Analysis"},{"metadata":{"_cell_guid":"7c6b9709-a895-43b2-831b-5b6bc2e3ca87","_uuid":"04a5e7b424751df95279eb538cdef5af3744e454","trusted":true,"collapsed":true},"cell_type":"code","source":"from sklearn import cross_validation\nfrom sklearn import tree\nfrom sklearn import svm\nfrom sklearn import ensemble\nfrom sklearn import neighbors\nfrom sklearn import linear_model\nfrom sklearn import metrics\nfrom sklearn import preprocessing","execution_count":4,"outputs":[]},{"metadata":{"_cell_guid":"df5b20b9-f18b-4480-9890-85ed4093dfff","_uuid":"dcd176e2bf9f7ba309216beed88f1a4361d70a2b","collapsed":true,"trusted":true},"cell_type":"code","source":"%matplotlib inline \n\nfrom IPython.display import Image\nimport matplotlib as mlp\nimport matplotlib.pyplot as plt\nimport numpy as np\nimport os\nimport pandas as pd\nimport sklearn\nimport seaborn as sns\n","execution_count":5,"outputs":[]},{"metadata":{"_cell_guid":"901d651c-c282-4fc9-8049-84f27dbe5065","_uuid":"1c307eb272a03428266b199cd4ac4505f211a7ea"},"cell_type":"markdown","source":"## Dataset \nhttps://www.kaggle.com/becksddf/churn-in-telecoms-dataset/data\n"},{"metadata":{"_cell_guid":"47bdcd89-d559-44a1-b9f0-62388921b324","_uuid":"0a3480ff88aeb7eca70cf58f493e7eefcc391aca","trusted":true},"cell_type":"code","source":"#df = pd.read_csv('../input/mytest.csv')\ndf = pd.read_csv('../input/bigml_59c28831336c6604c800002a.csv')\n\nprint (df.shape)\n\n#df.dtypes","execution_count":6,"outputs":[]},{"metadata":{"_cell_guid":"c1b7223d-10fa-4699-b6e1-59ebd0b04c1a","_uuid":"d3453663fe56554c122907f442f7eb2c73f96070","trusted":true},"cell_type":"code","source":"# Load data\ndf.head(3)","execution_count":7,"outputs":[]},{"metadata":{"_cell_guid":"cbe2d9d4-e0a7-4f91-90ec-f85cfa1d25de","_uuid":"332b0df4434f6e1e81935d6441600016401c2b31","trusted":true},"cell_type":"code","source":"y = df[\"churn\"].value_counts()\n#print (y)\nsns.barplot(y.index, y.values)","execution_count":8,"outputs":[]},{"metadata":{"_cell_guid":"42cbf1cd-8b39-4efb-a067-a011e7d9f19d","_uuid":"df4dd2a688a6bd520a222c71dcc8c245a4dcf3bb","trusted":true},"cell_type":"code","source":"y_True = df[\"churn\"][df[\"churn\"] == True]\nprint (\"Churn Percentage = \"+str((y_True.shape[0] / df[\"churn\"].shape[0]) * 100))","execution_count":9,"outputs":[]},{"metadata":{"_cell_guid":"a18bc10b-0730-4e1a-b92e-bd6031655783","_uuid":"9c048103b7607691f4f0400896a88670e07cb300"},"cell_type":"markdown","source":"### Conclusion 1 = Imbalanced data - Lesser datapoints in True Churn category"},{"metadata":{"_cell_guid":"db57eb70-ccb0-4cac-b15c-d7175b058152","_uuid":"6d905f747712b9d01e44207b06b248d4c7d4f6c3"},"cell_type":"markdown","source":"# Descriptive Analysis"},{"metadata":{"_cell_guid":"37bbe00f-cba0-4885-81b3-2f935ff9e007","_uuid":"18a23347fbad7024d9efb15ac7c0e9a83c645c74","trusted":true},"cell_type":"code","source":"\ndf.describe()","execution_count":10,"outputs":[]},{"metadata":{"_cell_guid":"fb5c8789-bc0c-4439-8ed5-a58daf68ce71","_uuid":"78e2978d435e9a10c568d50113703b813ce0ea15"},"cell_type":"markdown","source":"### Churn By State"},{"metadata":{"_cell_guid":"90d0411e-8a88-42ce-bbe2-7d201edfd716","_uuid":"590e158402076fdc683b4b8078b3b07bef060a85","trusted":true},"cell_type":"code","source":"df.groupby([\"state\", \"churn\"]).size().unstack().plot(kind='bar', stacked=True, figsize=(30,10))\n","execution_count":11,"outputs":[]},{"metadata":{"_cell_guid":"58c0b85d-26a6-44f2-be97-dabef60a6083","_uuid":"0b7e5ce03e5c6da5b31d28426c4b103b6e3291bc"},"cell_type":"markdown","source":"### Churn By Area Code"},{"metadata":{"_cell_guid":"098304e5-11cd-4977-88ff-134b58cb785b","_uuid":"68da3274171d6006e6da05c31a74d5ace44d120b","trusted":true},"cell_type":"code","source":"df.groupby([\"area code\", \"churn\"]).size().unstack().plot(kind='bar', stacked=True, figsize=(5,5))\n","execution_count":12,"outputs":[]},{"metadata":{"_cell_guid":"466abbf5-0923-4eb4-8a99-

```
9d1a4c8344fa","_uuid":"37617354d7a34411c0782165c5b6c6a9c80bb1b1"},"cell_type":"markdow
n","source":"### Churn By Customers with International
plan"},{"metadata":{"_cell_guid":"24a6ea96-e883-4de5-b9ba-
8d6f9c3a934a","_uuid":"aa434dd842477041da95c0e5068719da13398c7e","trusted":true},"cell
_type":"code","source":"df.groupby([\"international plan\",
\"churn\"]).size().unstack().plot(kind='bar', stacked=True, figsize=(5,5))
","execution_count":13,"outputs":[]},{"metadata":{"_cell_guid":"2a47ffd9-9838-40ea-
a6ab-
397baa434d4e","_uuid":"5420663aa16dc96695a6058832a9e0ac6348c6cc"},"cell_type":"markdow
n","source":"### Churn By Customers with Voice mail plan
"},{"metadata":{"_cell_guid":"7c6940b0-fbdc-4a97-8a87-
35429b6cf4fe","_uuid":"987bfc1de29f68622d327528a97cbe9d27cf4061","trusted":true},"cell
_type":"code","source":"df.groupby([\"voice mail plan\",
\"churn\"]).size().unstack().plot(kind='bar', stacked=True, figsize=(5,5))
","execution_count":14,"outputs":[]},{"metadata":{"_cell_guid":"a433d6c1-580e-4499-
b1db-
715e3a0ce3f6","_uuid":"e9fd0499ec7b8d51c9133d8b2f9cf53032123edf"},"cell_type":"markdow
n","source":"# Handle Categorical Cols - Label
Encode"},{"metadata":{"_cell_guid":"60686842-7b9e-4d1e-956a-
0383dd6aa1e5","_uuid":"0791c2aa13f06d384ea3edeb7ae46263ae70dbd7","collapsed":true,"tru
sted":true},"cell_type":"code","source":"# Discreet value integer
encoder\nlabel_encoder =
preprocessing.LabelEncoder()","execution_count":15,"outputs":[]},{"metadata":{"_cell_g
uid":"449cc6d8-1e4a-46c5-b3f9-
6e11efc679f2","_uuid":"e01a4d6c82d896cf4190bb010435c806295c3b92","scrolled":false,"tru
sted":true},"cell_type":"code","source":"# State is string and we want discreet
integer values\ndf['state'] =
label_encoder.fit_transform(df['state'])\ndf['international plan'] =
label_encoder.fit_transform(df['international plan'])\ndf['voice mail plan'] =
label_encoder.fit_transform(df['voice mail plan'])\n\n#print (df['Voice mail
plan'][:4])\nprint
(df.dtypes)","execution_count":16,"outputs":[]},{"metadata":{"_cell_guid":"72415434-
1578-4542-b5b5-
c211abcd4813","_uuid":"e0e500718a8167524390599e943920d421a51bc2","trusted":true},"cell
_type":"code","source":"df.shape","execution_count":17,"outputs":[]},{"metadata":{"_ce
ll_guid":"02357128-49ad-4c31-b7c5-
9042b4e3cbe4","_uuid":"3a0cb357fcb26db41f7a918ecf958adc6f087897","trusted":true},"cell
_type":"code","source":"df.head()","execution_count":18,"outputs":[]},{"metadata":{"_c
ell_guid":"3252b275-4dc8-47df-a517-
e759e50dfaad","_uuid":"621fbf18b62b229e3a08a48e355c95abf90e7d23"},"cell_type":"markdow
n","source":"### Strip of Response values"},{"metadata":{"_cell_guid":"67b04047-639a-
4b3d-b034-
15252e523833","_uuid":"123d6ecdd0205b238a09794d776c9cefa0cf0fde","trusted":true},"cell
_type":"code","source":"y =
df['churn'].as_matrix().astype(np.int)\ny.size","execution_count":19,"outputs":[]},{"m
etadata":{"_cell_guid":"725b00bd-6d6e-4d82-a7f5-
1eb4f00aa4de","_uuid":"98f7694085c4fedf3b5bb55013a41c3747eb15a8"},"cell_type":"markdow
n","source":"### Strip off Redundant cols"},{"metadata":{"_cell_guid":"e1bd8195-e8a2-
43ee-be9b-
8a743ec04b2a","_uuid":"dab41be8e68be4d250d49e67ac6f3f27a8440fc0","collapsed":true,"tru
sted":true},"cell_type":"code","source":"# df = df.drop([\"Id\",\"Churn\"], axis = 1,
inplace=True)\ndf.drop([\"phone number\",\"churn\"], axis = 1,
inplace=True)","execution_count":20,"outputs":[]},{"metadata":{"_cell_guid":"2e95129f-
a389-41b0-bd51-58567760602a","_kg_hide-input":false,"_kg_hide-
output":false,"_uuid":"165d5f885d79337773f49d11eef5e938b7e2ba59","trusted":true},"cell
_type":"code","source":"df.head(3)","execution_count":21,"outputs":[]},{"metadata":{"_
cell_guid":"39386f53-ed18-420e-b6ec-
f116a5495246","_uuid":"e1da5e5cd567651a66179dbdb9c9edba0b591f03"},"cell_type":"markdow
n","source":"### Build Feature Matrix"},{"metadata":{"_cell_guid":"fdc6ef47-46be-4a35-
b7ca-
97d4d5210d0a","_uuid":"ea6659f92330270cb0e8c704f4a9372066292897","collapsed":true,
"trusted":true},"cell_type":"code","source":"X =
df.as_matrix().astype(np.float)","execution_count":22,"outputs":[]},{"metadata":{"_cel
l_guid":"9aeb61c6-b6e2-4d31-9ca7-
4360e1db2a28","_uuid":"e326eca8ec77014a18a47137ccac6f39884caf3a","trusted":true},"cell
_type":"code","source":"X","execution_count":23,"outputs":[]},{"metadata":{"_cell_guid
```

```
":"d5243b57-af0a-4c3a-9242-
1c7f20a7a9cd","_uuid":"2dd19f8b942498c9e6c935d02b02865269cc1464","trusted":true},"cell
_type":"code","source":"X.shape","execution_count":24,"outputs":[]},{"metadata":{"_cel
l_guid":"79949972-fab1-4787-b456-
c21923ab54ae","_uuid":"216dc293983dccf8428edbe955626d5b93be1b33","collapsed":true},"ce
ll_type":"markdown","source":"### Standardize Feature Matrix
values"},{"metadata":{"_cell_guid":"52847a97-6c8a-476d-83c9-
a008c98edd44","_uuid":"cb8984ca0ef6d09b52aa64647abbc7720a777edc","collapsed":true,"tru
sted":true},"cell_type":"code","source":"scaler = preprocessing.StandardScaler()\nX =
scaler.fit_transform(X)","execution_count":25,"outputs":[]},{"metadata":{"_cell_guid":
"1a263d4c-ea1d-4b7b-a1a7-
1a8c427337f2","_uuid":"f306048cefd584f73dbce175e592fcfa282b43ce","trusted":true},"cell
_type":"code","source":"X","execution_count":26,"outputs":[]},{"metadata":{"_cell_guid
":"965a4e4e-2ed0-4f4a-9e61-
75a3e24e8715","_uuid":"6cb8b518410dccfbfadef42d0bdba775ba97a61a","collapsed":true,"tru
sted":true},"cell_type":"code","source":"","execution_count":null,"outputs":[]},{"meta
data":{"_cell_guid":"30d00638-5352-4e80-b075-
5336e145ff6d","_uuid":"39714e04e000a32ac712b1ed4c17c23958298ad5"},"cell_type":"markdow
n","source":"### Stratified Cross Validation - Since the Response values are not
balanced"},{"metadata":{"_cell_guid":"475e6f87-3e17-4b9c-8a61-
1f2a2221addf","_uuid":"dd67120e2892ea8ba3a71e517286a7f598fbaf4a","collapsed":true,"tru
sted":true},"cell_type":"code","source":"def stratified_cv(X, y, clf_class,
shuffle=True, n_folds=10, **kwargs):\n    stratified_k_fold =
cross_validation.StratifiedKFold(y, n_folds=n_folds, shuffle=shuffle)\n    y_pred =
y.copy()\n    # ii -> train\n    # jj -> test indices\n    for ii, jj in
stratified_k_fold: \n        X_train, X_test = X[ii], X[jj]\n        y_train = y[ii]\n
clf = clf_class(**kwargs)\n        clf.fit(X_train,y_train)\n        y_pred[jj] =
clf.predict(X_test)\n    return
y_pred","execution_count":27,"outputs":[]},{"metadata":{"_cell_guid":"f04e2b75-a019-
4b40-bc4b-
606a70b3d18f","_uuid":"11a2d46c102d7ff05b1f97c0f10e43a0f833f910","collapsed":true},"ce
ll_type":"markdown","source":"### Build Models and
Train"},{"metadata":{"_cell_guid":"a463a62e-93c1-4ca0-ae3b-
723523b8d5b0","_uuid":"454e2833bb27702062f888a9ce0efb8135f985f9","trusted":true},"cell
_type":"code","source":"print('Gradient Boosting Classifier:
{:.2f}'.format(metrics.accuracy_score(y, stratified_cv(X, y,
ensemble.GradientBoostingClassifier))))\nprint('Support vector machine(SVM):
{:.2f}'.format(metrics.accuracy_score(y, stratified_cv(X, y,
svm.SVC))))\nprint('Random Forest Classifier:
{:.2f}'.format(metrics.accuracy_score(y, stratified_cv(X, y,
ensemble.RandomForestClassifier))))\nprint('K Nearest Neighbor Classifier:
{:.2f}'.format(metrics.accuracy_score(y, stratified_cv(X, y,
neighbors.KNeighborsClassifier))))\nprint('Logistic Regression:
{:.2f}'.format(metrics.accuracy_score(y, stratified_cv(X, y,
linear_model.LogisticRegression))))","execution_count":28,"outputs":[]},{"metadata":{"
_cell_guid":"f56c778e-c908-4fc5-97a2-
e5450b9a6b14","_uuid":"e0521867fcf79031d9c36de34c83ff8cb146aca7","collapsed":true,"tru
sted":true},"cell_type":"code","source":"","execution_count":null,"outputs":[]},{"meta
data":{"_cell_guid":"ffbe3e5c-5c27-4b7b-a553-
e4262e5ce6fb","_uuid":"04d43d8fcbb190eade50b4de51171daa116e946a"},"cell_type":"markdow
n","source":"### Confusion Matrices for various
models"},{"metadata":{"_cell_guid":"0e58445b-3334-4261-9d91-
03b04bbdcedf","_uuid":"8c6b783e179539d67545b3cca4d2aa9551059171","trusted":true},"cell
_type":"code","source":"grad_ens_conf_matrix      = metrics.confusion_matrix(y,
stratified_cv(X, y, ensemble.GradientBoostingClassifier))\nsns.heatmap(grad_en
s_conf_matrix, annot=True,  fmt='');\ntitle = 'Gradient
Boosting'\nplt.title(title);","execution_count":29,"outputs":[]},{"metadata":{"_cell_g
uid":"06605a0e-6d6d-4955-b7c6-
12d9a5dc58ad","_uuid":"629fb1fd1d092aebd338b4bc65b3c88530ab5471","collapsed":true,"tru
sted":true},"cell_type":"code","source":"","execution_count":null,"outputs":[]},{"meta
data":{"_cell_guid":"e3a143d5-6506-4470-b68f-
c36a61084058","_uuid":"4ab168efde576acc28367de8c30d50f0f3e344ff","trusted":true},"cell
_type":"code","source":"svm_svc_conf_matrix       = metrics.confusion_matrix(y,
stratified_cv(X, y, svm.SVC))\nsns.heatmap(svm_svc_conf_matrix, annot=True,
fmt='');\ntitle =
'SVM'\nplt.title(title);","execution_count":30,"outputs":[]},{"metadata":{"_cell_guid"
```

```
:"9547e54a-d0fb-4793-9c2d-
b42d96e49d80","_uuid":"a98864ca689d21ac68ec20fe76d68fa6d13466f2","trusted":true},"cell
_type":"code","source":"random_forest_conf_matrix = metrics.confusion_matrix(y,
stratified_cv(X, y,
ensemble.RandomForestClassifier))\nsns.heatmap(random_forest_conf_matrix, annot=True,
fmt='');\ntitle = 'Random
Forest'\nplt.title(title);","execution_count":31,"outputs":[]},{"metadata":{"_cell_gui
d":"8a5546e2-7e80-45c7-9e20-
a2e016528746","_uuid":"c4e22980a7e7a729254c5f301a97d52af02c8897","trusted":true},"cell
_type":"code","source":"k_neighbors_conf_matrix   = metrics.confusion_matrix(y,
stratified_cv(X, y,
neighbors.KNeighborsClassifier))\nsns.heatmap(k_neighbors_conf_matrix, annot=True,
fmt='');\ntitle =
'KNN'\nplt.title(title);","execution_count":32,"outputs":[]},{"metadata":{"_cell_guid"
:"6dd055d6-d5e7-4f8c-a4ad-
2d8461d91c71","_uuid":"12ce331d2962d5b8ead17f756b4916743f9c21cb","trusted":true},"cell
_type":"code","source":"logistic_reg_conf_matrix  = metrics.confusion_matrix(y,
stratified_cv(X, y,
linear_model.LogisticRegression))\nsns.heatmap(logistic_reg_conf_matrix, annot=True,
fmt='');\ntitle = 'Logistic
Regression'\nplt.title(title);","execution_count":33,"outputs":[]},{"metadata":{"_cell
_guid":"ccbed6d4-6e11-4f38-b7b4-
a0e2d402318a","_uuid":"40ad782af75e023aa24c3b1cc05e6f926d314ff6"},"cell_type":"markdow
n","source":"### classification_report"},{"metadata":{"_cell_guid":"233ca82a-b951-
4642-9ebf-
ee07442e3b44","_uuid":"84f3861c18efcdcfa44d9e5cd41cf350bf7223e5","trusted":true},"cell
_type":"code","source":"print('Gradient Boosting Classifier:\\n
{}\\n'.format(metrics.classification_report(y, stratified_cv(X, y,
ensemble.GradientBoostingClassifier))))\nprint('Support vector machine(SVM):\\n
{}\\n'.format(metrics.classification_report(y, stratified_cv(X, y,
svm.SVC))))\nprint('Random Forest Classifier:\\n
{}\\n'.format(metrics.classification_report(y, stratified_cv(X, y,
ensemble.RandomForestClassifier))))\nprint('K Nearest Neighbor Classifier:\\n
{}\\n'.format(metrics.classification_report(y, stratified_cv(X, y,
neighbors.KNeighborsClassifier))))\nprint('Logistic Regression:\\n
{}\\n'.format(metrics.classification_report(y, stratified_cv(X, y,
linear_model.LogisticRegression))))","execution_count":34,"outputs":[]},{"metadata":{"
_cell_guid":"c4261223-5213-4d9c-8c35-
7912c94965f5","_uuid":"891aef6065cfcd655fba671a3675dc6a867848b0"},"cell_type":"markdow
n","source":"### Final Model Selection\n\nGradient Boosting seems to do comparatively
for this case"},{"metadata":{"_cell_guid":"299cf090-6cc9-4d06-bddc-
c7eacb7af6e6","_uuid":"3c8ab3e8869af22dd8030f78870e71edb068ec4f","trusted":true},"cell
_type":"code","source":"gbc = ensemble.GradientBoostingClassifier()\ngbc.fit(X,
y)","execution_count":35,"outputs":[]},{"metadata":{"_cell_guid":"6e878e57-c7a2-48da-
b218-
51c8d69d2088","_uuid":"d31627ac80203a4fe253e0bc08d1b3df00f4e81a","trusted":true},"cell
_type":"code","source":"# Get Feature Importance from the
classifier\nfeature_importance = gbc.feature_importances_\nprint
(gbc.feature_importances_)\nfeat_importances = pd.Series(gbc.feature_importances_,
index=df.columns)\nfeat_importances =
feat_importances.nlargest(19)\nfeat_importances.plot(kind='barh' , figsize=(10,10))
","execution_count":36,"outputs":[]},{"metadata":{"_cell_guid":"76f8a15c-91b4-4ef7-
aa4a-19c6dfcb5fc7","_uuid":"59e2e5a3e8f99f0c697cf36
1b3ad61a83b6d4acb","collapsed":true,"trusted":true},"cell_type":"code","source":"","ex
ecution_count":null,"outputs":[]},{"metadata":{"_cell_guid":"5a9caaca-4d8d-4a8d-9025-
7c8b7624d148","_uuid":"e418c7b5f674f0d2b8e2704f55e197ec52cba45f","collapsed":true,"tru
sted":true},"cell_type":"code","source":"","execution_count":null,"outputs":[]},{"meta
data":{"_cell_guid":"7f2dc7a5-26c6-4ca7-b81f-
bb747a34f264","_uuid":"10e191af9ae97e92aac0615594221ad660506308","collapsed":true,"tru
sted":true},"cell_type":"code","source":"","execution_count":null,"outputs":[]},{"meta
data":{"_cell_guid":"a5b5e4c5-e8b3-44bf-9782-
43d5a075ec98","_uuid":"0d48751e0fd6e4eed5e4395b4dd26d0b19a7e6ba","collapsed":true,"tru
sted":true},"cell_type":"code","source":"","execution_count":null,"outputs":[]}],"meta
data":{"anaconda-cloud":{},"kernelspec":{"display_name":"Python
3","language":"python","name":"python3"},"language_info":{"name":"python","version":"3
.6.5","mimetype":"text/x-
```

python","codemirror_mode":{"name":"ipython","version":3},"pygments_lexer":"ipython3","
nbconvert_exporter":"python","file_extension":".py"}},"nbformat":4,"nbformat_minor":1}