Carbon Monoxide Monitor Report

Nick Harbeck, Jacob King, Ben Thompson

1. Introduction.

Carbon Monoxide or CO poisoning is a dangerous condition that can cause serious tissue damage, or even death.[1] Many U.S. households are at risk of CO poisoning due to the use of fuel-burning appliances inside the home. This project uncovers the trends and risk factors associated with CO poisoning in U.S. households. During our analysis, we found that household income, rent, educational attainment, household heat source, and the year the household was constructed were the most significant predictors of CO monitor status.

To date, very little research has been performed on the analysis of homes at risk of CO poisoning. The U.S. Consumer Product Safety Commission (CPSC) provides several resources on the risks of CO poisoning[2] and this report provides additional empirical analysis of the CO poisoning risk in U.S. households.

A CPSC memorandum on the risks of CO poisoning found that CO monitors reduce the risk of death in a CO poisoning incident.[3] Conversely, households without monitors are at a much higher risk of death from CO poisoning. This report helps to prioritize constituent groups most at risk of CO poisoning which could be used for targeted policies that reduce the risk of CO poisoning.

We specifically analyze 14 predictors of CO monitor status in a U.S. household (present or not present) from the American Housing Survey. A combination of classification algorithms was used to predict the presence of a CO monitor in a U.S. household. This report contains detailed information on the dataset, data mining process, experimental setup, results, and opportunities for future research.

2. Description of the data set.

Our dataset is the U.S. Census Bureau's 2017 American Housing Survey (AHS). The AHS is a housing survey that collects information on a wide range of housing subjects, including characteristics of U.S. housing inventory.[4] The dataset contains over 66,000 responses from U.S. households and it contains features about the type of heating fuel used in a home, the presence of a CO monitor, demographic data, geographic information, and over 3000 other topics.[5] The data itself is encoded according to a predefined data dictionary.

[1] Carbon monoxide poisoning. 2019. Mayo Clinic. Accessed from https://www.mayoclinic.org/diseases-conditions/carbon-monoxide/symptoms-causes/syc-20370642
[2] Carbon Monoxide Information Center. 2019. United States Consumer Product Safety Commission. Accessed from https://www.cpsc.gov/Safety-Education/Safety-Education-Centers/Carbon-Monoxide-Information-Center
[3] Ault, Kimberly. Estimates of Non-fire Carbon Monoxide Poisoning Deaths and Injuries. 1997. Accessed from https://www.cpsc.gov/s3fs-public/pdfs/foia_3512c1f.pdf
[4] About this Survey. 2019. American Housing Survey (AHS). U.S. Census Bureau. Accessed from https://www.census.gov/programs-surveys/ahs/about.html
[5] The full 2017 AHS dataset can be accessed at https://www.census.gov/programs-surveys/ahs/data/2017/ahs-2017-public-use-file--puf-/ahs-2017-national-public-use-file--puf-.html

A subset of the full dataset was used for this analysis. 18 columns from the original dataset were used in this report and 14 of those columns were used to predict the target of CO monitor status. These 14 features are:

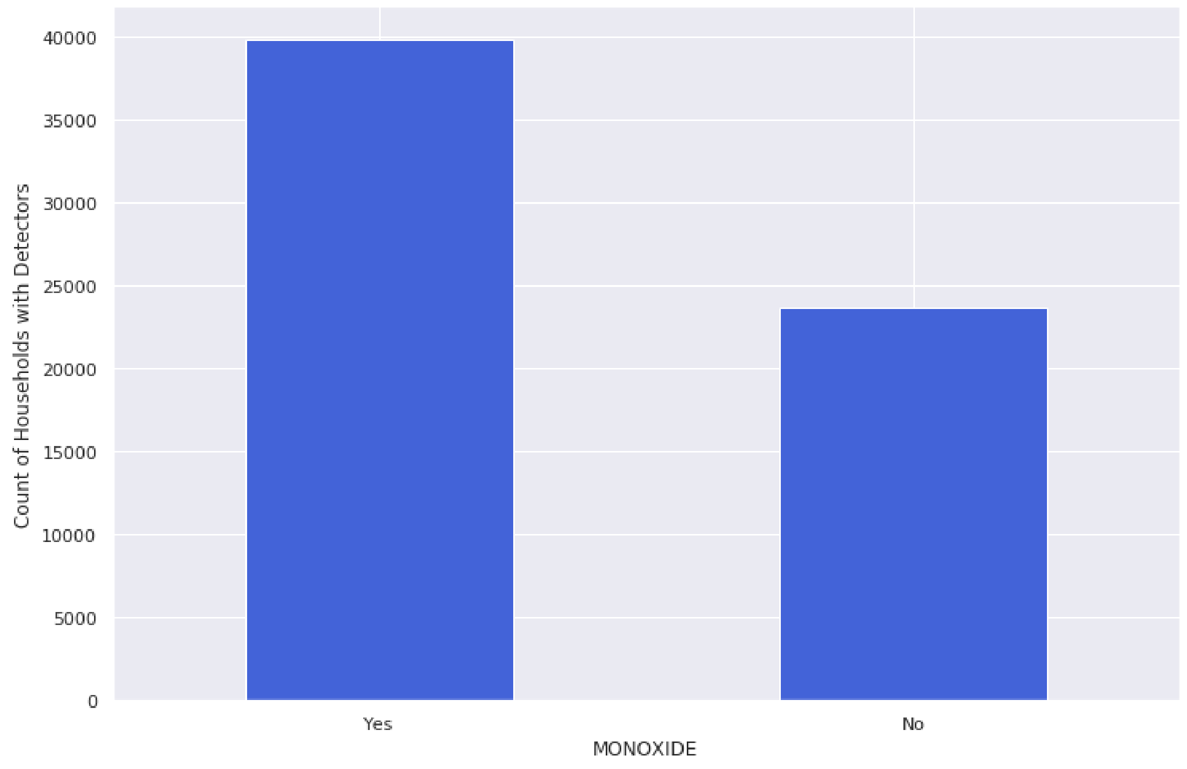
- Primary air conditioning system
- Secondary air conditioning system
- Type of heating equipment
- Fuel of heating equipment
- Hot water system
- Household income
- Homeowners association status (HOA)
- Year the household was built
- Rent payment
- Rent control status
- Marital status
- Educational attainment
- Race
- Household language

Our dataset also contained geographic location, UUID, and weighting factor of each sample which could be used in future analysis. These features were selected from the over 3000 original topics because they provide a good summary of key demographic factors and household indicators that could be associated with carbon monoxide status. The household indicators were selected because they are tied to the heating and cooling equipment in a household and gas appliances naturally produce CO as a byproduct of combustion.

3. Algorithm Description

Logistic regression, random forest, and support vector machine classifiers were used to analyze the dataset. This is because our labels for CO monitor status are not present or present. Initial exploration of the data found that most households have a CO monitor, but a significant portion do not have one.

Most U.S. Households have a CO Monitor



Our analysis found that certain features, including household income, rent status, year household was built, education level of the occupants, and the type of heating appliances are linked to the presence of a carbon monoxide monitor in U.S. household.

Python was the main tool to perform this analysis because it efficiently handles large datasets and a significant number of packages are available for data science work.

4. Experimental setup.

The first part of the data preprocessing was performed in R. The original AHS dataset was subset into the variables analyzed in this report. Next the file was cleaned in python by changing all responses to numerical data and encoding NA values correctly (the survey includes an NA response as a negative number that changes value depending on the question and response).

The three classifiers that we used were logistic regression, random forest, and support vector machine. We wanted to use these three techniques because they take different approaches to classification which could help increase the robustness of our results.

Due to the ease in deploying the models (CO status has two options; present or not present). We can use the classification report, accuracy, and misclassification rate to judge the performance of the models after the data is split into training and testing sets and 5-fold cross-validation.

For each classifier, the data was split into features and target and oversampled:

```
model1target = "MONOXIDE"
X1 = df_model1.drop(columns=[model1target]).values
y1 = df_model1[model1target].values
# RandomOverSampler (with random_state=0)
ros = RandomOverSampler(random_state=0)
X1, y1 = ros.fit_sample(X1, y1)
```

The data was cross validated using a function that was included in the full scikit-learn pipeline of each classifier:

```
def cvf(pipe,X,y, n_splits):
    accs = cross_val_score(pipe,
                           X,
                           y,
                           cv=KFold(n_splits=n_splits, random_state=0))
    print('The average accuracy score of the model is ', round(accs.
mean(), 3))
    print('The std deviation of the accuracy score is ', round(accs.
std(), 3))
    return round(accs.mean(), 3)
```

Each model was also scaled and placed into the pipeline:

```
pipe_logit = Pipeline([('StandardScaler', ss), ('Logistic', logit)])
```

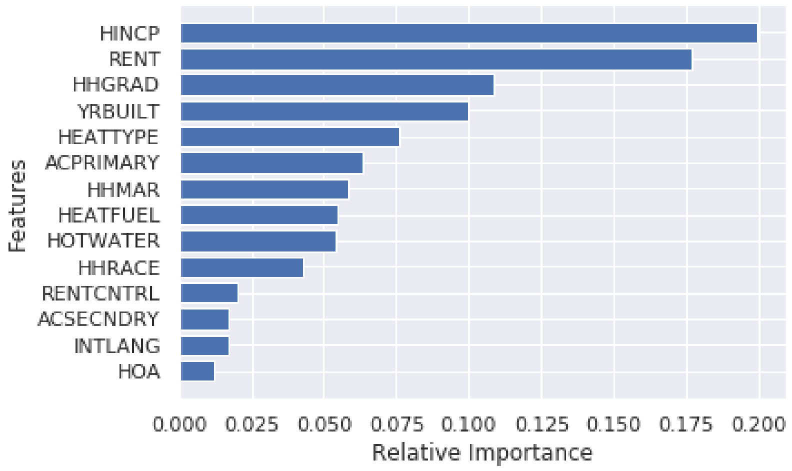
```
pipe_rf = Pipeline([('StandardScaler',ss), ('Random Forest', RandomF
orestClassifier(n_estimators=100, max_depth=10, min_samples_leaf=10,
 random_state=0))])
pipe_svm = Pipeline([('StandardScaler',ss), ('SVC', SVC(random_state
=0))])
```

The cross validated data models each ran, and we obtained the following accuracies:

```
model Logistic ...
The average accuracy score of the model is  0.182
The std deviation of the accuracy score is  0.094
0:00:00.085932

 model Random Forest ...
The average accuracy score of the model is  0.401
The std deviation of the accuracy score is  0.057
0:00:01.671271

 model SVC ...
The average accuracy score of the model is  0.333
The std deviation of the accuracy score is  0.066
0:00:04.589659

 ------------------------------
accuracy  -  model with Kfolds=5 cross-validation
0.401 - Random Forest
0.333 - SVC
0.182 - Logistic
```

We then used this information to obtain the most important features associated with the random forest model (highest accuracy model).

According to our model, the most important features are income, rent status, education level, heat type, and year built



To improve accuracy, we tried running an ensemble of all three models combined using scikit-learn VotingClassifier (a bagging method), but the model's accuracy did not increase.

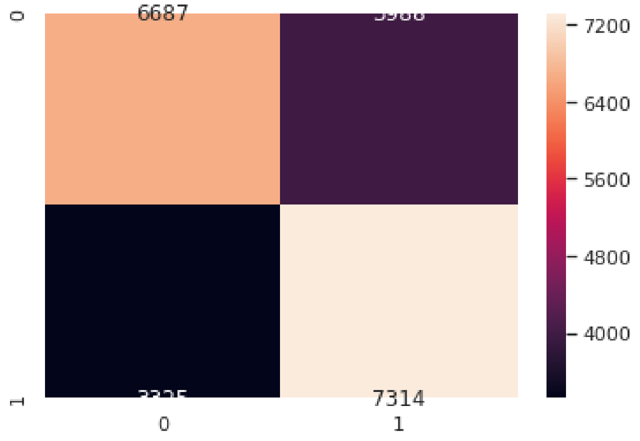
```
model Vote ...
The average accuracy score of the model is  0.285
The std deviation of the accuracy score is  0.076
0:00:06.538192

 ------------------------------
accuracy  -  model with Kfolds=5 cross-validation
0.285 - Vote
```

We were not happy with the results of the logistic regression and random forest classification models, so we continued to analyze the dataset using these classifiers on only the five most important features.

```
 model Logistic ...
The average accuracy score of the model is  0.566
The std deviation of the accuracy score is  0.066
0:00:00.335734

 model Random Forest ...
The average accuracy score of the model is  0.622
The std deviation of the accuracy score is  0.05
0:00:06.958918

 model SVC ...
The average accuracy score of the model is  0.614
The std deviation of the accuracy score is  0.043
0:02:26.778303

 ------------------------------
accuracy  -  model with Kfolds=5 cross-validation
0.622 - Random Forest
0.614 - SVC
0.566 - Logistic
```

The new models are not perfect, but they are much more accurate. We can also use the classification report and confusion matrix to analyze the performance of random forest – our most accurate model.

```
              precision    recall  f1-score   support

           1       0.67      0.63      0.65     10675
           2       0.65      0.69      0.67     10639

    accuracy                           0.66     21314
   macro avg       0.66      0.66      0.66     21314
weighted avg       0.66      0.66      0.66     21314
```



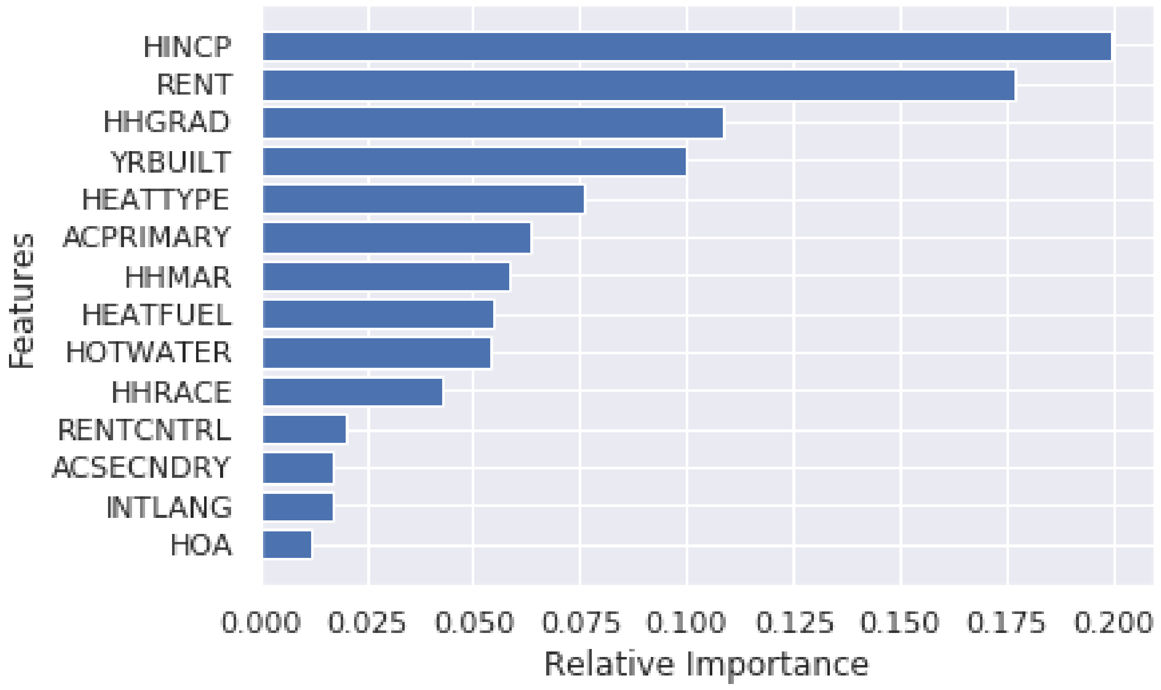
5. Results.

Our results reflect the three major models that we ran; logistic regression, support vector machine, and random forest, after analyzing only the five most important features. The random forest model had the highest accuracy and best classification report according to F1-score, misclassification rate, and accuracy. Even when ensembling the three models together, we did not see any significant increase in model performance. For interpretability of our results, we decided to analyze the original three models. Logistic regression also performed moderately well, and the combination of logistic regression and random forest results provided several key insights.

Random forest models can display feature importance which is helpful for determining the most important features of our dataset according to relative importance. Logistic regression feature coefficients can also be referenced to determine how variability of the most important features affect the outcome of CO monitor status.

As referenced earlier, household income, rent, educational attainment, household heat source, and the year the household was constructed were the most significant predictors of CO monitor status.

According to our model, the most important features are income, rent status, education level, year built, and heat type



In fact, when running the model with household income as the only predictor, we found that accuracy was approximately 58%.

For each of these features, we used a logistic regression to obtain the coefficients of each response option. This allows us to see how the most important features, as indicated by the random forest model, affect the likelihood of CO monitor presence in a household.

We can see how the features change the likelihood of CO monitor presence in a household below:

```
HOTWATER     -6.446435e-02
ACSECNDRY    -4.420390e-02
HEATFUEL     -3.032724e-02
HHGRAD       -1.845504e-02
HOA          -3.016613e-03
HHMAR        -8.478037e-04
RENT         -1.140057e-04
HINCP         5.251367e-07
YRBUILT       6.651862e-04
INTLANG       6.890235e-03
ACPRIMARY     1.730811e-02
RENTCNTRL     1.740987e-02
HHRACE        3.376590e-02
HEATTYPE      3.546891e-02
```

Higher income, gas-fueled appliances, lower rent, older homes, and lower education levels increase the likelihood of a household with a CO monitor. Some of these results are interesting because we expected older homes, lower rent, and lower education households to be less likely to have a CO monitor because these features tend to be correlated with lower household income.[6] This is an area that would be a good direction for future analysis

6. Summary and conclusions.

CO monitor status in U.S. households can be predicted from a variety of factors listed in the 2017 AHS. Our results allow us to conclude that any efforts to increase the presence of CO monitors in U.S. households should target low income households first. This is because household income is the most important feature according to our random forest classifier and lower income households are less likely than higher income households to have a CO monitor according to the logistic regression model.

We also saw some counterintuitive results in the models that we ran. For example, lower rent, and lower education households all tend to increase the likelihood of a household having a CO monitor. The lower rent and graduate status could also be features that are closely correlated with households that have gas appliances. If there is no source of gas combustion (in the case of higher rent and higher education status households), CO generation, a byproduct of gas combustion, is much less likely to occur in a home and the importance of having a CO monitor diminishes.

The 2017 AHS dataset offers many opportunities for data analysis and mining. In the future, it would be helpful to look at different models, optimize model hyperparameters, and expand the scope of features that can be used to predict the presence of a CO monitor in a U.S. household. Further research could also be performed to combine survey results with CO incidents in the U.S. The Federal Emergency Management Administration (FEMA) reports chemical release incident data for the U.S. and lists incidents according to chemical type and location.[7] The geographic indicators in the 2017 AHS could be combined with the FEMA data to look at how CO poisoning severity links to demographic and other appliance factors as listed in the AHS dataset.

[6] Douglas-Hall, Ayana and Chau, Michelle. 2007. Parents' Low Education Leads to Low Income, Despite Full-Time Employment. Accessed from http://www.nccp.org/publications/pub_786.html
[7] National Fire Incident Reporting System. Accessed from https://www.nfirs.fema.gov/

Appendix of Computer Listings (Code for this report only, no GUI)

```
# import packages
import pandas as pd
from datetime import datetime
import numpy as np
import re
import warnings
from sklearn.exceptions import DataConversionWarning
warnings.filterwarnings("ignore", category=DeprecationWarning)
warnings.filterwarnings(action='ignore', category=DataConversionWarning)

#The below file was pared down into important features from the full 2017 American
Housing Survey dataset. The code for this can be found in R syntax in the Github
repo.
COdf = pd.read_csv("https://nickharbeck.com/wp-content/uploads/2019/11/AHSCO.csv")
#The next few lines of code set all strings as numbers.
for col in
COdf[['MONOXIDE','CONTROL','HEATTYPE','HEATFUEL','HOTWATER','ACPRIMARY','ACSECNDRY','
HOA','RENTCNTRL','HHMAR','HHGRAD','HHRACE','INTLANG']]:
  COdf[col] = COdf[col].apply(lambda x: re.findall('[^0-9-]+([0-9-]+)', str(x)))
  COdf[col] = COdf[col].str.get(0)
  COdf[col] = pd.to_numeric(COdf[col])

#Here's a clean and smaller copy of the labels hosted on my site
AHS_LABELSNH = pd.read_csv("https://nickharbeck.com/wp-
content/uploads/2019/11/ValueLabels.csv")
AHS_LABELSNH['Value'] = AHS_LABELSNH['Value'].str.extract('(\d+)', expand=False)

#The cleaned dataset is a copy of the original dataset that includes complete rows
df = COdf.fillna(-10)
df = df[(df['MONOXIDE']>= 0 ) & (df['HOA']>=0) & (df['YRBUILT']>=0) & (df['RENT']>=0)
& (df['RENTCNTRL']>=0) & (df['HHMAR']>=0) & (df['HHGRAD']>=0) & (df['HHRACE']>=0) &
(df['INTLANG']>=0) & (df['WEIGHT']>=0) & (df['HEATTYPE'] >= 0) & (df['HEATFUEL']>= 0
) & (df['HOTWATER']>=0) & (df['ACPRIMARY']>=0) & (df['ACPRIMARY']>=0) &
(df['ACSECNDRY']>=0) & (df['HINCP']>= 0)]

print("The highest income in the dataset is $",COdf['HINCP'].max(), "per year.")
print("The lowest income in the dataset is $", df['HINCP'].min(), "per year.")
df.dtypes

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
from sklearn.pipeline import Pipeline
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
```

```
ss = StandardScaler()
logit = LogisticRegression(random_state=0, solver='liblinear')

import matplotlib.pyplot as plt
import seaborn as sns
def col_histogram(df,column):
    plt.figure(figsize=(12,8))
    sns.set(style="darkgrid")
    sns.distplot(df[column].values)
    plt.tight_layout()
    plt.show()

features =
['MONOXIDE','CONTROL','HEATTYPE','HEATFUEL','HOTWATER','ACPRIMARY','ACSECNDRY','HOA',
'RENTCNTRL','HHMAR','HHGRAD','HHRACE','INTLANG']
for i in features:
  col_histogram(df,i)

detectorscount = COdf[(COdf['MONOXIDE']>= 0 )]
detectorscount = detectorscount.groupby('MONOXIDE').MONOXIDE.count()
fig,ax = plt.subplots(figsize = (12,8))
detectorscount.plot(kind='bar', color = '#4363d8')
plt.ylabel('Count of Households with Detectors')
plt.xticks(np.arange(2),('Yes','No'), rotation = 'horizontal')
plt.show()

HINCPModel = COdf[(COdf['MONOXIDE']>= 0 )&(COdf['HINCP']>= 0 )]
HINCPModel = HINCPModel[['MONOXIDE','HINCP']]
X = HINCPModel['HINCP'].values
y = HINCPModel['MONOXIDE'].values

le = LabelEncoder()
y = le.fit_transform(y)
from imblearn.over_sampling import RandomOverSampler
y = y.reshape(-1,1)
X = X.reshape(-1,1)

# RandomOverSampler (with random_state=0)
# Implement me
ros = RandomOverSampler(random_state=0)
X, y = ros.fit_sample(X, y)

pd.DataFrame(data=y, columns=['MONOXIDE'])['MONOXIDE'].value_counts()

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.8,
random_state=0)
X_train = ss.fit_transform(X_train.reshape(-1, 1))
X_test = ss.transform(X_test.reshape(-1,1))

logit_model = logit.fit(X_train,y_train)
y_predicted = logit_model.predict(X_test)
accuracy_score(y_test, y_predicted)

from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
print(confusion_matrix(y_test, y_predicted))
print(classification_report(y_test,y_predicted))

df_model1 = COdf[(COdf['MONOXIDE']>= 0 ) & (COdf['HOA']>=0) & (COdf['YRBUILT']>=0) &
(COdf['RENT']>=0) & (COdf['RENTCNTRL']>=0) & (COdf['HHMAR']>=0) & (COdf['HHGRAD']>=0)
& (COdf['HHRACE']>=0) & (COdf['INTLANG']>=0) & (COdf['WEIGHT']>=0) &
(COdf['HEATTYPE'] >= 0) & (COdf['HEATFUEL']>= 0 ) & (COdf['HOTWATER']>=0) &
(COdf['ACPRIMARY']>=0) & (COdf['ACPRIMARY']>=0) & (COdf['ACSECNDRY']>=0) &
(COdf['HINCP']>= 0)]
df_model1 = df_model1.drop(columns = ["CONTROL", "OMB13CBSA", "WEIGHT"])
model1target = "MONOXIDE"
X1 = df_model1.drop(columns=[model1target]).values
y1 = df_model1[model1target].values

# RandomOverSampler (with random_state=0)
ros = RandomOverSampler(random_state=0)
X1, y1 = ros.fit_sample(X1, y1)

def cvf(pipe,X,y, n_splits):
    accs = cross_val_score(pipe,
                           X,
                           y,
                           cv=KFold(n_splits=n_splits, random_state=0))
    print('The average accuracy score of the model is ', round(accs.mean(), 3))
    print('The std deviation of the accuracy score is ', round(accs.std(), 3))
    return round(accs.mean(), 3)

pipe_logit = Pipeline([('StandardScaler', ss), ('Logistic', logit)])
pipe_rf = Pipeline([('StandardScaler',ss), ('Random Forest',
RandomForestClassifier(n_estimators=100, max_depth=10, min_samples_leaf=10,
random_state=0))])
pipe_svm = Pipeline([('StandardScaler',ss), ('SVC', SVC(random_state=0))])

# prepare python dictionary with models to test
pipe_estimators_model1 = {}

pipe_estimators_model1['Logistic'] = pipe_logit
pipe_estimators_model1['Random Forest'] = pipe_rf
pipe_estimators_model1['SVC'] = pipe_svm

Kfolds = 5
scores = []

for name, pipe in pipe_estimators_model1.items():
    print('\n model {} ...'.format(name))
    #%time score = cvf(pipe, X1,y1,Kfolds)
    start_time = datetime.now()
    score = cvf(pipe, X1,y1,Kfolds)
    print(datetime.now()-start_time)
    scores.append((name, score))

print("\n ------------------------------")
sorted_scores = sorted(scores, key=lambda x: x[1], reverse=True)
print("accuracy  -  model with Kfolds={} cross-validation".format(Kfolds))
```

```
for score in sorted_scores:
    print("{:0.3f} - {}".format(score[1], score[0]))

Model1_dict = dict(zip(df_model1.drop(columns=[model1target]).columns,
pipe_rf.steps[1][1].fit(X1,y1).feature_importances_))

SortModel1 = pd.Series(Model1_dict).sort_values(ascending=True)
print(SortModel1)
SortModel1 = SortModel1.reset_index()

print('According to our random forest model, the most important features are income,
rent status, education level, heat type, and year built')
plt.xlabel("Relative Importance")
plt.ylabel('Features')
plt.yticks(range(len(list(SortModel1['index']))), list(SortModel1['index']))
ax.invert_yaxis()
importances=pd.Series(RandomForestClassifier(n_estimators=100, max_depth=10,
min_samples_leaf=10, random_state=0).fit(X1,y1).feature_importances_)
plt.barh(range(len(list(SortModel1['index']))), list(SortModel1[0]), align='center')
plt.show()

#We can also use logistic regression to see how the features change the likelihood of
CO monitor presence in a household

Model1v2_dict = dict(zip(df_model1.drop(columns=[model1target]).columns,
(pipe_logit.steps[1][1].fit(X1,y1).coef_).flat))
SortModel1v2 = pd.Series(Model1v2_dict).sort_values(ascending=True)
print(SortModel1v2)
print("As seen above, higher income, newer homes, lower rent, fuel-burning appliances
and lower education levels increase the likelihood of a household with a CO monitor")

#Now lets see if ensembling improves the accuracy
from sklearn.ensemble import VotingClassifier
pipe_ensemble = Pipeline([('StandardScaler', ss), ('Vote',
VotingClassifier(estimators=[('Logit', pipe_logit), ('rf', pipe_rf), ('svc',
pipe_svm)], voting='hard'))])
pipe_estimators_model2 = {}
pipe_estimators_model2['Vote'] = pipe_ensemble

Kfolds = 5
scores = []

for name, pipe in pipe_estimators_model2.items():
    print('\n model {} ...'.format(name))
    #%time score = cvf(pipe, X1,y1,Kfolds)
    start_time=datetime.now()
    score = cvf(pipe, X1,y1,Kfolds)
    print(datetime.now() - start_time)
    scores.append((name, score))

print("\n ------------------------------")
sorted_scores = sorted(scores, key=lambda x: x[1], reverse=True)
print("accuracy  -  model with Kfolds={} cross-validation".format(Kfolds))
```

```
for score in sorted_scores:
    print("{:0.3f} - {}".format(score[1], score[0]))

print('Since the ensembling method did not improve accuracy, we will analyze the
dataset using the models using only the five most important features')
df_model3 = COdf[(COdf['MONOXIDE']>= 0 ) & (COdf['YRBUILT']>=0) & (COdf['RENT']>=0) &
(COdf['HHGRAD']>=0) & (COdf['HEATTYPE'] >= 0) & (COdf['HINCP']>= 0)]
df_model3 = df_model3.drop(columns = ["CONTROL", "OMB13CBSA", "WEIGHT"])
model3target = "MONOXIDE"
X3 = df_model3.drop(columns=[model3target]).values
y3 = df_model3[model3target].values

# RandomOverSampler (with random_state=0)
ros = RandomOverSampler(random_state=0)
X3, y3 = ros.fit_sample(X3, y3)

Kfolds = 5
scores = []

pipe_estimators_model3 = pipe_estimators_model1

for name, pipe in pipe_estimators_model3.items():
    print('\n model {} ...'.format(name))
    #%time score = cvf(pipe, X1,y1,Kfolds)
    start_time = datetime.now()
    score = cvf(pipe, X3,y3,Kfolds)
    print(datetime.now()-start_time)
    scores.append((name, score))

print("\n -------------------------------")
sorted_scores = sorted(scores, key=lambda x: x[1], reverse=True)
print("accuracy  -  model with Kfolds={} cross-validation".format(Kfolds))

for score in sorted_scores:
    print("{:0.3f} - {}".format(score[1], score[0]))
```