Selezione delle Feature

Questo notebook si propone di indagare se sia possibile **ridurre lo spazio delle colonne** dei dataset d'interesse, ossia se sia possibile limitare il numero di feature (le feature sono eliminate perché ne esistono altre molto correlate e che portano un'informazione analoga che sarebbe ridondante, o perché contribuiscono con un'informazione particolarmente ridotta).

In [1]:
```python
# Libraries
import pandas            as pd
import numpy             as np
import matplotlib.pyplot as plt
import warnings

from sklearn.metrics           import mean_squared_error
from sklearn.ensemble          import RandomForestRegressor
from sklearn.utils             import resample
from sklearn.feature_selection import RFECV
from sklearn.model_selection   import cross_val_score

warnings.filterwarnings('ignore')
```

Lettura dei dataset divisi per contea.

In [2]:
```python
# local file paths

dir_name = 'contea'
region_names = np.array(['A', 'B', 'C'])
region_ids = ['1286', '2061', '3101']

fp_Xtrain = []
fp_Xval   = []
fp_Xtest  = []
fp_ytrain = []
fp_yval   = []
fp_ytest  = []

for i in range(3):
    fp_Xtrain.append(dir_name + f'/X_train{region_names[i]}.csv')
    fp_Xval  .append(dir_name + f'/X_val{  region_names[i]}.csv')
    fp_Xtest .append(dir_name + f'/X_test{ region_names[i]}.csv')
    fp_ytrain.append(dir_name + f'/y_train{region_names[i]}.csv')
    fp_yval  .append(dir_name + f'/y_val{  region_names[i]}.csv')
    fp_ytest .append(dir_name + f'/y_test{ region_names[i]}.csv')
```

In [3]:
```python
# Lettura dei dati

X_train = []
X_val   = []
X_test  = []
y_train = []
y_val   = []
y_test  = []

for i in range(3):
    X_train.append(pd.read_csv(fp_Xtrain[i], low_memory=False))
    X_val  .append(pd.read_csv(fp_Xval  [i], low_memory=False))
    X_test .append(pd.read_csv(fp_Xtest [i], low_memory=False))
    y_train.append(pd.read_csv(fp_ytrain[i], low_memory=False))
    y_val  .append(pd.read_csv(fp_yval  [i], low_memory=False))
    y_test .append(pd.read_csv(fp_ytest [i], low_memory=False))

X_train = np.array(X_train, dtype=object)
X_val   = np.array(X_val,   dtype=object)
X_test  = np.array(X_test,  dtype=object)
y_train = np.array(y_train, dtype=object)
y_val   = np.array(y_val,   dtype=object)
y_test  = np.array(y_test,  dtype=object)
```

In [4]:
```python
def dimensionality(y=False):
    for i in range(3):
        print(f'X_train{region_names[i]}: {X_train[i].shape}')
        print(f'X_val{region_names[i]}:   {X_val  [i].shape}')
        print(f'X_test{region_names[i]}:  {X_test [i].shape}')
        if y:
            print(f'y_train{region_names[i]}: {y_train[i].shape}')
            print(f'y_val{region_names[i]}:   {y_val  [i].shape}')
            print(f'y_test{region_names[i]}:  {y_test [i].shape}')
            print()
```

In [5]:
```python
dimensionality(y=True)
```

```
X_trainA: (26819, 69)
X_valA:   (9006, 69)
X_testA:  (9085, 69)
y_trainA: (26819, 1)
y_valA:   (9006, 1)
y_testA:  (9085, 1)

X_trainB: (8119, 69)
X_valB:   (2658, 69)
X_testB:  (2606, 69)
y_trainB: (8119, 1)
y_valB:   (2658, 1)
y_testB:  (2606, 1)

X_trainC: (64771, 69)
X_valC:   (21908, 69)
X_testC:  (21876, 69)
y_trainC: (64771, 1)
y_valC:   (21908, 1)
y_testC:  (21876, 1)
```

Definizione di un Subset per la costruzione del modello

Gli algoritmi usati in questo notebook hanno un costo computazionale elevato, per questo è definito un sottoinsieme dei dataset originali su cui far girare gli algoritmi; questo per facilitare la fase di creazione e di testing e ottenere risultati verosimili in tempi utili a verificare il corretto funzionamento del processo.

Nella versione finale gli algoritmi usaranno la totalità delle instanze del dataset originale.

In [6]:
```python
#sub_perc = [1/1000, 1/100, 1/1000] # A:   100, B:  100, C:   100; Test rapido      ~  3 min
#sub_perc = [   1/3,     1,    1/4] # A:  8000, B: 8000, C:  8000; Test medio       ~ 15 min
#sub_perc = [     1,     1,    2/5] # A: 24000, B: 8000, C: 24000; Test medio       ~  1 ora
sub_perc = [     1,     1,      1]  # A: 24000, B: 8000, C: 64000; Dataset Completo ~  4 ore e 30 min
```

In [7]:
```python
for i in range(3):
    print(int(len(X_train[i])*sub_perc[i]))
```

```
26819
8119
64771
```

In [8]:
```python
X_train_sub = []
y_train_sub = []
```

In [9]:
```python
for i in range(3):
    X_sub, y_sub = resample(X_train[i], y_train[i], n_samples = int(sub_perc[i]*len(X_train[i])))
    X_train_sub.append(X_sub)
    y_train_sub.append(y_sub)
```

Variabili globali

In [10]:
```python
# Globals
N_ESTIMATORS = 80
CV = 5
SCORING = 'neg_mean_squared_error'
N_JOBS  = -1
```

Importanza delle Feature: Random Forest

Il primo luogo è allenato un **RandomForestRegressor**, da cui si può ricavare il ranking delle feature per importanza (si fa supporto di un *barplot* per una rappresentazione grafica di questa informazione).

Sulla base del ranking ottenuto viene anche modellato il miglior modello per numero di feature usate sulla base del **mean squared error**: si itera sul numero di feature usate aggiunendone una alla volta in base al ranking. (Anche per questo tipo di informazione è fornito un grafico).

In [11]:
```python
plt.rcParams.update({'font.size': 35})
```

In [12]:
```python
def createRF(X, y):
    rf = RandomForestRegressor(
        n_estimators = N_ESTIMATORS,
        n_jobs = N_JOBS
    )
    rf.fit(X, y.values.ravel())
    return rf
```

In [13]:
```python
def barplot(X, rf, reg_name, file_name=''):

    print(f'{reg_name} FEATURE IMPORTANCES')
    print(list(X.columns[np.argsort(rf.feature_importances_)[::-1]]))
    print()

    fig, ax = plt.subplots(figsize=(len(rf.feature_importances_)/2,10))

    ax.tick_params(axis='x', which='major', labelsize=15)
    ax.tick_params(axis='x', which='minor', labelsize=20)
    ax.tick_params(axis='y', which='major', labelsize=25)
    ax.tick_params(axis='y', which='minor', labelsize=30)

    ax.bar(range(0, X.shape[1]), rf.feature_importances_)
    ax.set_title("Feature Importances")
    ax.set_xticks(range(X.shape[1]))
    ax.set_xticklabels(X.columns, rotation=90)

    ax.grid()

    if file_name != '':
        fig.savefig('images/' + file_name + '_all_feature_importances.jpg')
```

In [14]:
```python
def compute_rmse(X, y, bf, debug=False):

    rmse  = []

    for f in range(1, len(bf)+1):

        if debug:
            print(f)

        rf_small = RandomForestRegressor(
            n_estimators = N_ESTIMATORS,
            n_jobs = N_JOBS
        )
        cols     = X.columns[bf][:f]
        scores   = cross_val_score(
            rf_small,
            X.loc[:,cols],
            y.values.ravel(),
            cv = CV,
            scoring = SCORING,
        )
        rmse += [-scores.mean()]

    return rmse
```

In [15]:
```python
def feature_plot(rmse, reg_name, file_name=''):

    min_ = min(rmse)
    best = np.argmin(rmse) + 1

    print(f'{reg_name} INCREASING n-FEATURE PLOT')
    print ("Full score:", rmse[-1])
    print ("Best score:", min_)
    print("Best number of feature: ", best)
    print()

    fig, ax = plt.subplots(figsize=(len(rmse)/2, 10))

    ax.tick_params(axis='x', which='major', labelsize=25)
    ax.tick_params(axis='x', which='minor', labelsize=30)
    ax.tick_params(axis='y', which='major', labelsize=30)
    ax.tick_params(axis='y', which='minor', labelsize=35)

    ax.plot(range(1, len(rmse)+1), rmse, 'o-', label="RMSE")
    ax.set_title("RMSE on varying features")
    ax.set_xlabel("Number of Best features used")

    ax.grid()

    if file_name != '':
        fig.savefig('images/' + file_name + '_increasing_rmse.jpg')
```

In [16]:
```python
def feature_importance(X, y, reg_name, debug=False, file_name=''):

    rf = createRF(X, y)

    barplot(X, rf, reg_name, file_name=file_name)

    best_features = np.argsort(rf.feature_importances_)[::-1]

    rmse = compute_rmse(X, y, best_features, debug=debug)

    feature_plot(rmse, reg_name, file_name=file_name)
```

In [17]:
```python
def feat_importance(index, debug=False, file_name=''):
    if file_name == '':
        file_name = region_ids[index]
    return feature_importance(
        X_train_sub[index],
        y_train_sub[index],
        region_names[index],
        debug=debug,
        file_name=file_name
)
```

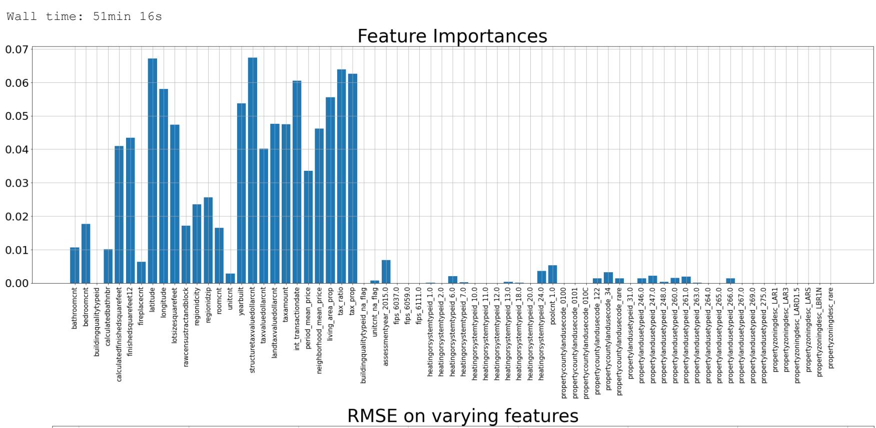
Prima Contea 1286

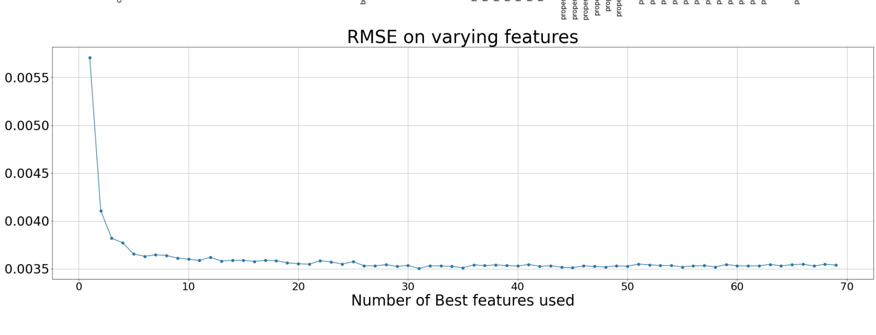
In [18]:
```python
%%time
feat_importance(
    0#,
    #file_name = 'prova_1'
)
```

```
A FEATURE IMPORTANCES
['structuretaxvaluedollarcnt', 'latitude', 'tax_ratio', 'tax_prop', 'int_transactiondate', 'longitude', 'living_area_prop', 'yearbuilt', 'landtaxvaluedollarcnt', 'taxamount', 'lotsizesquarefeet', 'neighborhood_mean_price', 'finishedsquarefeet12', 'calculatedfinishedsquarefeet', 'taxvaluedollarcnt', 'period_mean_price', 'regionidzip', 'regionidcity', 'bedroomcnt', 'rawcensustractandblock', 'roomcnt', 'bathroomcnt', 'calculatedbathnbr', 'assessmentyear_2015.0', 'fireplacecnt', 'poolcnt_1.0', 'heatingorsystemtypeid_24.0', 'propertycountylandusecode_34', 'unitcnt', 'propertylandusetypeid_247.0', 'heatingorsystemtypeid_6.0', 'propertylandusetypeid_261.0', 'propertylandusetypeid_260.0', 'propertylandusetypeid_246.0', 'propertycountylandusecode_rare', 'propertycountylandusecode_122', 'propertylandusetypeid_266.0', 'unitcnt_na_flag', 'heatingorsystemtypeid_13.0', 'propertylandusetypeid_248.0', 'heatingorsystemtypeid_7.0', 'heatingorsystemtypeid_18.0', 'heatingorsystemtypeid_1.0', 'propertylandusetypeid_263.0', 'heatingorsystemtypeid_10.0', 'propertylandusetypeid_269.0', 'heatingorsystemtypeid_11.0', 'propertycountylandusecode_0101', 'propertycountylandusecode_010C', 'propertyzoningdesc_LARS', 'buildingqualitytypeid', 'propertyzoningdesc_LARD1.5', 'propertyzoningdesc_LAR3', 'propertyzoningdesc_LAR1', 'propertylandusetypeid_275.0', 'propertyzoningdesc_LBR1N', 'propertylandusetypeid_267.0', 'heatingorsystemtypeid_2.0', 'propertylandusetypeid_265.0', 'propertylandusetypeid_264.0', 'fips_6111.0', 'fips_6059.0', 'fips_6037.0', 'heatingorsystemtypeid_12.0', 'buildingqualitytypeid_na_flag', 'propertylandusetypeid_31.0', 'heatingorsystemtypeid_20.0', 'propertycountylandusecode_0100', 'propertyzoningdesc_rare']

A INCREASING n-FEATURE PLOT
Full score: 0.0035387929552893084
Best score: 0.0035037512427053934
Best number of feature:  31

Wall time: 51min 16s
```





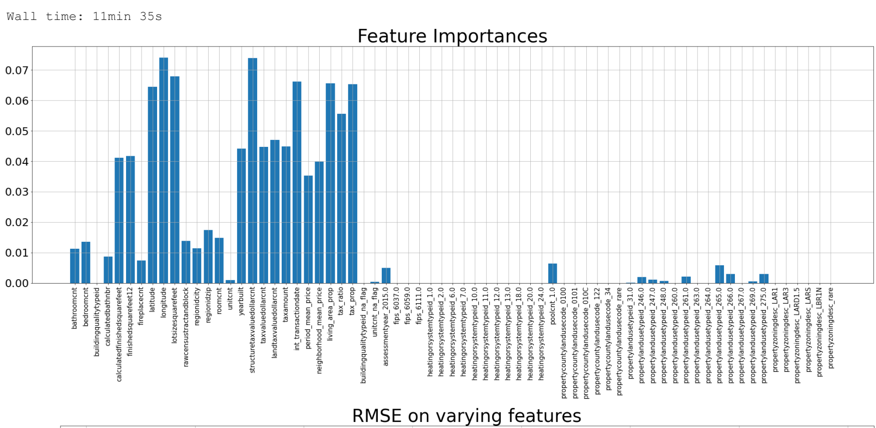
Seconda Contea 2061

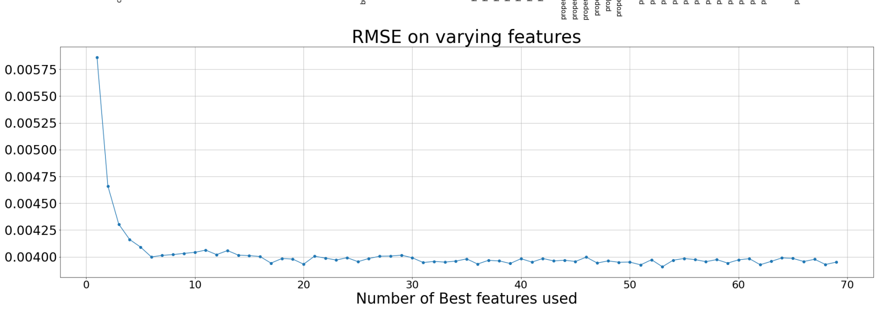
In [19]:
```python
%%time
feat_importance(
    1#,
    #file_name = 'prova_2'
)
```

```
B FEATURE IMPORTANCES
['longitude', 'structuretaxvaluedollarcnt', 'lotsizesquarefeet', 'int_transactiondate', 'living_area_prop', 'tax_prop', 'latitude', 'tax_ratio', 'landtaxvaluedollarcnt', 'taxamount', 'taxvaluedollarcnt', 'yearbuilt', 'finishedsquarefeet12', 'calculatedfinishedsquarefeet', 'neighborhood_mean_price', 'period_mean_price', 'regionidzip', 'roomcnt', 'rawcensustractandblock', 'bedroomcnt', 'regionidcity', 'bathroomcnt', 'calculatedbathnbr', 'fireplacecnt', 'poolcnt_1.0', 'propertylandusetypeid_265.0', 'assessmentyear_2015.0', 'propertylandusetypeid_275.0', 'propertylandusetypeid_266.0', 'propertylandusetypeid_261.0', 'propertylandusetypeid_246.0', 'propertylandusetypeid_247.0', 'unitcnt', 'propertylandusetypeid_248.0', 'propertylandusetypeid_269.0', 'unitcnt_na_flag', 'propertylandusetypeid_31.0', 'propertylandusetypeid_263.0', 'buildingqualitytypeid', 'buildingqualitytypeid_na_flag', 'propertyzoningdesc_rare', 'fips_6037.0', 'fips_6059.0', 'propertyzoningdesc_LARS', 'propertyzoningdesc_LARD1.5', 'propertyzoningdesc_LAR3', 'propertyzoningdesc_LAR1', 'propertylandusetypeid_267.0', 'propertylandusetypeid_264.0', 'propertylandusetypeid_260.0', 'propertycountylandusecode_rare', 'propertycountylandusecode_34', 'propertycountylandusecode_122', 'propertycountylandusecode_010C', 'propertycountylandusecode_0101', 'propertycountylandusecode_0100', 'heatingorsystemtypeid_24.0', 'heatingorsystemtypeid_20.0', 'heatingorsystemtypeid_18.0', 'heatingorsystemtypeid_13.0', 'heatingorsystemtypeid_12.0', 'heatingorsystemtypeid_11.0', 'heatingorsystemtypeid_10.0', 'heatingorsystemtypeid_7.0', 'propertyzoningdesc_LBR1N', 'heatingorsystemtypeid_2.0', 'heatingorsystemtypeid_1.0', 'fips_6111.0', 'heatingorsystemtypeid_6.0']

B INCREASING n-FEATURE PLOT
Full score: 0.003950691598601619
Best score: 0.00390728526283164
Best number of feature:  53

Wall time: 11min 35s
```





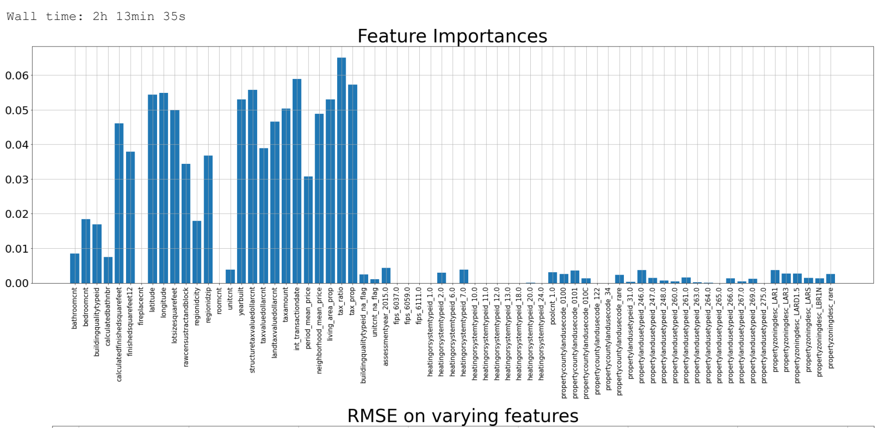
Terza Contea 3101

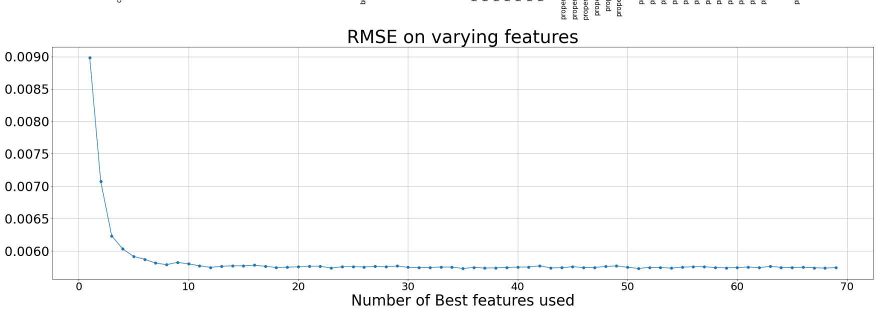
In [20]:
```python
%%time
feat_importance(
    2#,
    #file_name = 'prova_3'
)
```

```
C FEATURE IMPORTANCES
['tax_ratio', 'int_transactiondate', 'tax_prop', 'structuretaxvaluedollarcnt', 'longitude', 'latitude', 'living_area_prop', 'yearbuilt', 'taxamount', 'lotsizesquarefeet', 'neighborhood_mean_price', 'landtaxvaluedollarcnt', 'calculatedfinishedsquarefeet', 'taxvaluedollarcnt', 'finishedsquarefeet12', 'regionidzip', 'rawcensustractandblock', 'period_mean_price', 'bedroomcnt', 'regionidcity', 'buildingqualitytypeid', 'bathroomcnt', 'calculatedbathnbr', 'assessmentyear_2015.0', 'unitcnt', 'heatingorsystemtypeid_7.0', 'propertylandusetypeid_246.0', 'propertyzoningdesc_LAR1', 'propertycountylandusecode_0101', 'poolcnt_1.0', 'heatingorsystemtypeid_2.0', 'propertyzoningdesc_LARD1.5', 'propertyzoningdesc_LAR3', 'propertyzoningdesc_rare', 'propertycountylandusecode_0100', 'buildingqualitytypeid_na_flag', 'propertycountylandusecode_rare', 'propertylandusetypeid_261.0', 'propertylandusetypeid_247.0', 'propertyzoningdesc_LARS', 'propertycountylandusecode_010C', 'propertyzoningdesc_LBR1N', 'propertylandusetypeid_266.0', 'propertylandusetypeid_269.0', 'unitcnt_na_flag', 'propertylandusetypeid_248.0', 'propertylandusetypeid_260.0', 'propertylandusetypeid_267.0', 'propertylandusetypeid_31.0', 'propertylandusetypeid_263.0', 'heatingorsystemtypeid_20.0', 'propertylandusetypeid_264.0', 'fireplacecnt', 'propertylandusetypeid_275.0', 'propertylandusetypeid_265.0', 'propertycountylandusecode_34', 'roomcnt', 'heatingorsystemtypeid_10.0', 'propertycountylandusecode_122', 'heatingorsystemtypeid_24.0', 'heatingorsystemtypeid_18.0', 'heatingorsystemtypeid_13.0', 'heatingorsystemtypeid_12.0', 'heatingorsystemtypeid_11.0', 'fips_6037.0', 'fips_6059.0', 'fips_6111.0', 'heatingorsystemtypeid_1.0', 'heatingorsystemtypeid_6.0']

C INCREASING n-FEATURE PLOT
Full score: 0.005743334310014274
Best score: 0.0057313560862826355
Best number of feature:  51

Wall time: 2h 13min 35s
```





Analisi dei risultati

Il comportamento è simile per tutte le contee, seppur per la prima il miglior *mean squared error* si ottenga con una trentina di feature mentre per la seconda e terza il numero ottimale sia di circa 50, l'andamento e il valore finale dell'errore è comparabile.
Per tutte e tre l'importanza delle variabili è analoga:

- `latitude` e `longitude` sono molto importanti, segno che la posizione geografica ha una certa rilevanza.
- la colonna `int_transactiondate` ha una gran rilevanza, sintomo che il periodo di ventina nell'anno è molto significativo.
- le colonne relative alle tasse sono quelle che spiccano per importanza, probabilmente perché grandezze che includono più fattori, tra cui spiccano le due colonne aggiunte in fase di preparazione `tax_ratio` e `tax_prop`.
- le altre tre colonne sintetiche `period_mean_price`, `neighborhood_mean_price` e `living_area_prop` sono tra le più importanti.
- tutte le feature generate dal One-Hot-Encoding di `fips` e `heatingorsystemtypeid` non sono importanti.
- invece feature generate dal One-Hot-Encoding come `propertylandusetypeid` o `propertyzoningdesc` hanno se sommate nel complesso una certa importanza.

Importanza delle feature: Selezione Ricorsiva

Il **ranking delle feature** attraverso la RandomForest non è del tutto affidabile: due variabili molto correlate sono in competizione per portare lo stesso tipo di informazione, se questa informazione fosse elevata entrambe avrebbero uno score medio; sarebbe invece preferibile avere un unica feature con uno score molto alto.
Per permettere ciò utilizzo una **selezione ricorsiva 1-step**: ad ogni passo rialleno un albero e scarto la feature con importanza minore. Questo processo permette di evitare il rischio prima descritto: eliminando una delle due feature che sono fortemente correlate e che potrebbero avere uno score medio, all'iterazione successiva avendo un'unica feature questa avrà uno score molto alto.

In [21]:
```python
def get_selector(X, y):
    rf_small = RandomForestRegressor(
        n_estimators = N_ESTIMATORS,
        n_jobs = N_JOBS
    )
    selector = RFECV(
        rf_small,
        step    = 1,
        cv      = CV,
        scoring = SCORING,
        min_features_to_select = 1,
        n_jobs   = N_JOBS
    )
    selector.fit(X, y.values.ravel())
    return selector
```

In [22]:
```python
def selector_info(sel, X):

    print("SELECTOR")
    print("Numero di Feature selezionate: ", sel.n_features_)
    print("Feature: \n", list(X.columns[sel.support_]))
    print()
    print("Ranking delle feature: \n", list(X.columns[np.argsort(sel.ranking_)][:sel.n_features_]))
    print()

    return list(X.columns[np.argsort(sel.ranking_)][:sel.n_features_])
```

In [23]:
```python
def recursive_selection(X, y):

    selector = get_selector(X, y)

    selected = selector_info(selector, X)

    return selected
```

In [24]:
```python
feat_selected = []
```

In [25]:
```python
def get_selected(index):
    return recursive_selection(
        X_train_sub[index],
        y_train_sub[index]
)
```

Prima Contea 1286

In [26]:
```python
%%time
feat_selected.append(get_selected(0))
```

```
SELECTOR
Numero di Feature selezionate:  55
Feature:
 ['bathroomcnt', 'bedroomcnt', 'buildingqualitytypeid', 'calculatedbathnbr', 'calculatedfinishedsquarefeet', 'finishedsquarefeet12', 'fireplacecnt', 'latitude', 'longitude', 'lotsizesquarefeet', 'rawcensustractandblock', 'regionidcity', 'regionidzip', 'roomcnt', 'unitcnt', 'yearbuilt', 'structuretaxvaluedollarcnt', 'taxvaluedollarcnt', 'landtaxvaluedollarcnt', 'taxamount', 'int_transactiondate', 'period_mean_price', 'neighborhood_mean_price', 'living_area_prop', 'tax_ratio', 'tax_prop', 'buildingqualitytypeid_na_flag', 'unitcnt_na_flag', 'assessmentyear_2015.0', 'heatingorsystemtypeid_1.0', 'heatingorsystemtypeid_6.0', 'heatingorsystemtypeid_7.0', 'heatingorsystemtypeid_10.0', 'heatingorsystemtypeid_11.0', 'heatingorsystemtypeid_12.0', 'heatingorsystemtypeid_13.0', 'heatingorsystemtypeid_18.0', 'heatingorsystemtypeid_24.0', 'poolcnt_1.0', 'propertycountylandusecode_0100', 'propertycountylandusecode_122', 'propertycountylandusecode_34', 'propertycountylandusecode_rare', 'propertylandusetypeid_31.0', 'propertylandusetypeid_246.0', 'propertylandusetypeid_247.0', 'propertylandusetypeid_248.0', 'propertylandusetypeid_260.0', 'propertylandusetypeid_261.0', 'propertylandusetypeid_263.0', 'propertylandusetypeid_264.0', 'propertylandusetypeid_265.0', 'propertylandusetypeid_266.0', 'propertylandusetypeid_267.0', 'propertylandusetypeid_269.0']

Ranking delle feature:
 ['bathroomcnt', 'heatingorsystemtypeid_7.0', 'heatingorsystemtypeid_10.0', 'heatingorsystemtypeid_11.0', 'heatingorsystemtypeid_12.0', 'heatingorsystemtypeid_13.0', 'heatingorsystemtypeid_18.0', 'heatingorsystemtypeid_24.0', 'poolcnt_1.0', 'propertycountylandusecode_0100', 'propertycountylandusecode_122', 'propertycountylandusecode_34', 'heatingorsystemtypeid_1.0', 'propertycountylandusecode_rare', 'propertylandusetypeid_246.0', 'propertylandusetypeid_247.0', 'propertylandusetypeid_248.0', 'propertylandusetypeid_260.0', 'propertylandusetypeid_261.0', 'propertylandusetypeid_263.0', 'propertylandusetypeid_264.0', 'propertylandusetypeid_265.0', 'propertylandusetypeid_266.0', 'propertylandusetypeid_267.0', 'propertylandusetypeid_269.0', 'propertylandusetypeid_31.0', 'assessmentyear_2015.0', 'heatingorsystemtypeid_6.0', 'buildingqualitytypeid_na_flag', 'bedroomcnt', 'buildingqualitytypeid', 'calculatedbathnbr', 'calculatedfinishedsquarefeet', 'finishedsquarefeet12', 'fireplacecnt', 'latitude', 'longitude', 'lotsizesquarefeet', 'rawcensustractandblock', 'regionidcity', 'regionidzip', 'roomcnt', 'unitcnt_na_flag', 'yearbuilt', 'unitcnt', 'tax_ratio', 'tax_prop', 'taxvaluedollarcnt', 'landtaxvaluedollarcnt', 'taxamount', 'int_transactiondate', 'period_mean_price', 'neighborhood_mean_price', 'living_area_prop', 'structuretaxvaluedollarcnt']

Wall time: 52min 28s
```

Seconda Contea 2061

In [27]:
```python
%%time
feat_selected.append(get_selected(1))
```

```
SELECTOR
Numero di Feature selezionate:  32
Feature:
 ['bathroomcnt', 'bedroomcnt', 'calculatedbathnbr', 'calculatedfinishedsquarefeet', 'finishedsquarefeet12', 'fireplacecnt', 'latitude', 'longitude', 'lotsizesquarefeet', 'rawcensustractandblock', 'regionidcity', 'regionidzip', 'roomcnt', 'unitcnt', 'yearbuilt', 'structuretaxvaluedollarcnt', 'taxvaluedollarcnt', 'landtaxvaluedollarcnt', 'taxamount', 'int_transactiondate', 'period_mean_price', 'neighborhood_mean_price', 'living_area_prop', 'tax_ratio', 'tax_prop', 'assessmentyear_2015.0', 'poolcnt_1.0', 'propertylandusetypeid_246.0', 'propertylandusetypeid_261.0', 'propertylandusetypeid_265.0', 'propertylandusetypeid_266.0', 'propertylandusetypeid_275.0']

Ranking delle feature:
 ['bathroomcnt', 'propertylandusetypeid_246.0', 'poolcnt_1.0', 'propertylandusetypeid_265.0', 'propertylandusetypeid_266.0', 'propertylandusetypeid_275.0', 'propertylandusetypeid_261.0', 'tax_prop', 'tax_ratio', 'living_area_prop', 'neighborhood_mean_price', 'period_mean_price', 'int_transactiondate', 'taxamount', 'landtaxvaluedollarcnt', 'taxvaluedollarcnt', 'assessmentyear_2015.0', 'yearbuilt', 'calculatedfinishedsquarefeet', 'finishedsquarefeet12', 'calculatedbathnbr', 'bedroomcnt', 'fireplacecnt', 'latitude', 'structuretaxvaluedollarcnt', 'lotsizesquarefeet', 'rawcensustractandblock', 'regionidcity', 'regionidzip', 'roomcnt', 'unitcnt', 'longitude']

Wall time: 12min 46s
```

Terza Contea 3101

In [28]:
```python
%%time
feat_selected.append(get_selected(2))
```

```
SELECTOR
Numero di Feature selezionate:  33
Feature:
 ['bathroomcnt', 'bedroomcnt', 'buildingqualitytypeid', 'calculatedbathnbr', 'calculatedfinishedsquarefeet', 'finishedsquarefeet12', 'latitude', 'longitude', 'lotsizesquarefeet', 'rawcensustractandblock', 'regionidcity', 'regionidzip', 'unitcnt', 'yearbuilt', 'structuretaxvaluedollarcnt', 'taxvaluedollarcnt', 'landtaxvaluedollarcnt', 'taxamount', 'int_transactiondate', 'period_mean_price', 'neighborhood_mean_price', 'living_area_prop', 'tax_ratio', 'tax_prop', 'assessmentyear_2015.0', 'heatingorsystemtypeid_2.0', 'heatingorsystemtypeid_7.0', 'propertycountylandusecode_0100', 'propertycountylandusecode_0101', 'propertycountylandusecode_rare', 'propertylandusetypeid_246.0', 'propertyzoningdesc_LAR1', 'propertyzoningdesc_rare']

Ranking delle feature:
 ['bathroomcnt', 'int_transactiondate', 'period_mean_price', 'neighborhood_mean_price', 'living_area_prop', 'tax_ratio', 'tax_prop', 'taxamount', 'assessmentyear_2015.0', 'heatingorsystemtypeid_7.0', 'propertycountylandusecode_0100', 'propertycountylandusecode_0101', 'propertycountylandusecode_rare', 'propertylandusetypeid_246.0', 'propertyzoningdesc_LAR1', 'heatingorsystemtypeid_2.0', 'landtaxvaluedollarcnt', 'propertyzoningdesc_rare', 'structuretaxvaluedollarcnt', 'yearbuilt', 'unitcnt', 'bedroomcnt', 'regionidzip', 'regionidcity', 'rawcensustractandblock', 'lotsizesquarefeet', 'buildingqualitytypeid', 'calculatedbathnbr', 'longitude', 'calculatedfinishedsquarefeet', 'finishedsquarefeet12', 'taxvaluedollarcnt', 'latitude']

Wall time: 2h 38min 57s
```

Analisi dei risultati

In [29]:
```python
def print_list_info(l):
    print(*l, sep='\n')
    print(f'({len(l)})')
```

Prima Contea 1286

In [30]:
```python
print_list_info(feat_selected[0])
```

```
bathroomcnt
heatingorsystemtypeid_7.0
heatingorsystemtypeid_10.0
heatingorsystemtypeid_11.0
heatingorsystemtypeid_12.0
heatingorsystemtypeid_13.0
heatingorsystemtypeid_18.0
heatingorsystemtypeid_24.0
poolcnt_1.0
propertycountylandusecode_0100
propertycountylandusecode_122
propertycountylandusecode_34
heatingorsystemtypeid_1.0
propertycountylandusecode_rare
propertylandusetypeid_246.0
propertylandusetypeid_247.0
propertylandusetypeid_248.0
propertylandusetypeid_260.0
propertylandusetypeid_261.0
propertylandusetypeid_263.0
propertylandusetypeid_264.0
propertylandusetypeid_265.0
propertylandusetypeid_266.0
propertylandusetypeid_267.0
propertylandusetypeid_269.0
propertylandusetypeid_31.0
assessmentyear_2015.0
heatingorsystemtypeid_6.0
buildingqualitytypeid_na_flag
bedroomcnt
buildingqualitytypeid
calculatedbathnbr
calculatedfinishedsquarefeet
finishedsquarefeet12
fireplacecnt
latitude
longitude
lotsizesquarefeet
rawcensustractandblock
regionidcity
regionidzip
roomcnt
unitcnt_na_flag
yearbuilt
unitcnt
tax_ratio
tax_prop
taxvaluedollarcnt
landtaxvaluedollarcnt
taxamount
int_transactiondate
period_mean_price
neighborhood_mean_price
living_area_prop
structuretaxvaluedollarcnt
(55)
```

Seconda Contea 2061

In [31]:
```python
print_list_info(feat_selected[1])
```

```
bathroomcnt
propertylandusetypeid_246.0
poolcnt_1.0
propertylandusetypeid_265.0
propertylandusetypeid_266.0
propertylandusetypeid_275.0
propertylandusetypeid_261.0
tax_prop
tax_ratio
living_area_prop
neighborhood_mean_price
period_mean_price
int_transactiondate
taxamount
landtaxvaluedollarcnt
taxvaluedollarcnt
assessmentyear_2015.0
yearbuilt
calculatedfinishedsquarefeet
finishedsquarefeet12
calculatedbathnbr
bedroomcnt
fireplacecnt
latitude
structuretaxvaluedollarcnt
lotsizesquarefeet
rawcensustractandblock
regionidcity
regionidzip
roomcnt
unitcnt
longitude
(32)
```

Terza Contea 3101

In [32]:
```python
print_list_info(feat_selected[2])
```

```
bathroomcnt
int_transactiondate
period_mean_price
neighborhood_mean_price
living_area_prop
tax_ratio
tax_prop
taxamount
assessmentyear_2015.0
heatingorsystemtypeid_7.0
propertycountylandusecode_0100
propertycountylandusecode_0101
propertycountylandusecode_rare
propertylandusetypeid_246.0
propertyzoningdesc_LAR1
heatingorsystemtypeid_2.0
landtaxvaluedollarcnt
propertyzoningdesc_rare
structuretaxvaluedollarcnt
yearbuilt
unitcnt
bedroomcnt
regionidzip
regionidcity
rawcensustractandblock
lotsizesquarefeet
buildingqualitytypeid
calculatedbathnbr
longitude
calculatedfinishedsquarefeet
finishedsquarefeet12
taxvaluedollarcnt
latitude
(33)
```

Colonne da rimuovere:

In [33]:
```python
to_delete = []
for i in range(3):
    to_del = list(set(X_train[i].columns) - set(feat_selected[i]))
    to_delete.append(to_del)
```

Prima Contea 1286

In [34]:
```python
print_list_info(to_delete[0])
```

```
propertyzoningdesc_LBR1N
propertyzoningdesc_LAR1
propertyzoningdesc_LARS
propertycountylandusecode_010C
propertyzoningdesc_rare
fips_6037.0
propertyzoningdesc_LARD1.5
heatingorsystemtypeid_20.0
propertylandusetypeid_275.0
propertycountylandusecode_0101
fips_6059.0
fips_6111.0
heatingorsystemtypeid_2.0
propertyzoningdesc_LAR3
(14)
```

Seconda Contea 2061

In [35]:
```python
print_list_info(to_delete[1])
```

```
propertylandusetypeid_267.0
heatingorsystemtypeid_24.0
propertycountylandusecode_0100
propertyzoningdesc_LAR1
heatingorsystemtypeid_1.0
propertycountylandusecode_34
propertycountylandusecode_122
fips_6037.0
propertylandusetypeid_260.0
propertyzoningdesc_LARD1.5
heatingorsystemtypeid_6.0
heatingorsystemtypeid_11.0
heatingorsystemtypeid_12.0
propertylandusetypeid_248.0
heatingorsystemtypeid_13.0
heatingorsystemtypeid_18.0
fips_6111.0
heatingorsystemtypeid_2.0
propertyzoningdesc_LAR3
propertyzoningdesc_LBR1N
heatingorsystemtypeid_10.0
propertylandusetypeid_264.0
buildingqualitytypeid
heatingorsystemtypeid_20.0
propertycountylandusecode_rare
fips_6059.0
propertylandusetypeid_269.0
propertylandusetypeid_247.0
propertyzoningdesc_rare
unitcnt_na_flag
propertylandusetypeid_263.0
propertylandusetypeid_31.0
propertyzoningdesc_LARS
propertycountylandusecode_010C
heatingorsystemtypeid_7.0
propertycountylandusecode_0101
buildingqualitytypeid_na_flag
 (37)
```

Terza Contea 3101

In [36]:

```python
print_list_info(to_delete[2])
```

```
propertylandusetypeid_267.0
heatingorsystemtypeid_24.0
heatingorsystemtypeid_1.0
propertylandusetypeid_265.0
propertylandusetypeid_266.0
propertycountylandusecode_34
propertycountylandusecode_122
fips_6037.0
propertylandusetypeid_260.0
propertyzoningdesc_LARD1.5
poolcnt_1.0
heatingorsystemtypeid_6.0
heatingorsystemtypeid_11.0
heatingorsystemtypeid_12.0
propertylandusetypeid_248.0
heatingorsystemtypeid_13.0
heatingorsystemtypeid_18.0
propertylandusetypeid_275.0
fips_6111.0
propertyzoningdesc_LAR3
propertyzoningdesc_LBR1N
heatingorsystemtypeid_10.0
propertylandusetypeid_264.0
heatingorsystemtypeid_20.0
fips_6059.0
propertylandusetypeid_269.0
propertylandusetypeid_247.0
unitcnt_na_flag
propertylandusetypeid_263.0
propertylandusetypeid_31.0
propertyzoningdesc_LARS
propertycountylandusecode_010C
roomcnt
fireplacecnt
propertylandusetypeid_261.0
buildingqualitytypeid_na_flag
 (36)
```

Sembra che principalmente siano eliminate la maggior parte delle colonne generate dal One-Hot Encoding, queste sembrano avere scarsa importanza come evidenziava anche la prima analisi legata al ranking di una foresta.

In [37]:

```python
def remove_column(df, col_names):
    df.drop(col_names, axis=1, inplace=True)
    return df
```

In [38]:

```python
dimensionality(y=True)
```

```
X_trainA: (26819, 69)
X_valA:   (9006, 69)
X_testA:  (9085, 69)
y_trainA: (26819, 1)
y_valA:   (9006, 1)
y_testA:  (9085, 1)

X_trainB: (8119, 69)
X_valB:   (2658, 69)
X_testB:  (2606, 69)
y_trainB: (8119, 1)
y_valB:   (2658, 1)
y_testB:  (2606, 1)

X_trainC: (64771, 69)
X_valC:   (21908, 69)
X_testC:  (21876, 69)
y_trainC: (64771, 1)
y_valC:   (21908, 1)
y_testC:  (21876, 1)
```

In [39]:

```python
for i in range(3):
    for X in [X_train[i], X_val[i], X_test[i]]:
        X = remove_column(X, to_delete[i])
```

In [40]:

```python
dimensionality(y=True)
```

```
X_trainA: (26819, 55)
X_valA:   (9006, 55)
X_testA:  (9085, 55)
y_trainA: (26819, 1)
y_valA:   (9006, 1)
y_testA:  (9085, 1)

X_trainB: (8119, 32)
X_valB:   (2658, 32)
X_testB:  (2606, 32)
y_trainB: (8119, 1)
y_valB:   (2658, 1)
y_testB:  (2606, 1)

X_trainC: (64771, 33)
X_valC:   (21908, 33)
X_testC:  (21876, 33)
y_trainC: (64771, 1)
y_valC:   (21908, 1)
y_testC:  (21876, 1)
```

Analisi dei risultati

Il nuemero di feature selezionate è analogo a quello ottenuto dall'increasing RMSE: una cinquantina di feature per la prima contea e una trentina per la seconda e la terza. Analisi delle feature comuni.

Feature selezionate comuni a tutte e tre le contee:

In [41]:

```python
common = set(X_train[0].columns)

for i in range(1,3):
    common = common.intersection(set(X_train[i].columns))
common = list(common)

print_list_info(common)
```

```
taxvaluedollarcnt
latitude
structuretaxvaluedollarcnt
tax_prop
finishedsquarefeet12
rawcensustractandblock
regionidcity
yearbuilt
bedroomcnt
lotsizesquarefeet
propertylandusetypeid_246.0
assessmentyear_2015.0
taxamount
longitude
calculatedfinishedsquarefeet
unitcnt
living_area_prop
bathroomcnt
neighborhood_mean_price
int_transactiondate
landtaxvaluedollarcnt
calculatedbathnbr
period_mean_price
tax_ratio
regionidzip
 (25)
```

Le feature più importanti sono le stesse individuate anche dal ranking della foresta: `longitude` e `latitude` per l'area geografica, `int_tansactiondate` per il periodo di vendita e tutte le feature legate alle tasse, che sono probabilmente capaci di considerare in un unica variabili diversi aspetti dell'abitazione.

Tutte le colonne aggiunte in fase di preparazione (`int_transactiondate` , `period_mean_price` , `neighborhood_mean_price` , `tax_prop` e `living_area_prop`) sono state selezionate.

Scrittura dei dati

Salvo i dataset su cui è stata fatta la selezione delle feature su una specifica cartella.

In [42]:

```python
dir_name = 'selezione'

for i in range(3):
    X_train[i].to_csv( dir_name + f'/X_train{region_names[i]}.csv',  index=False)
    X_val  [i].to_csv( dir_name + f'/X_val{  region_names[i]}.csv',  index=False)
    X_test [i].to_csv( dir_name + f'/X_test{ region_names[i]}.csv',  index=False)
    y_train[i].to_csv( dir_name + f'/y_train{region_names[i]}.csv',  index=False)
    y_val  [i].to_csv( dir_name + f'/y_val{  region_names[i]}.csv',  index=False)
    y_test [i].to_csv( dir_name + f'/y_test{ region_names[i]}.csv',  index=False)
```