Dry – Data Preparation:

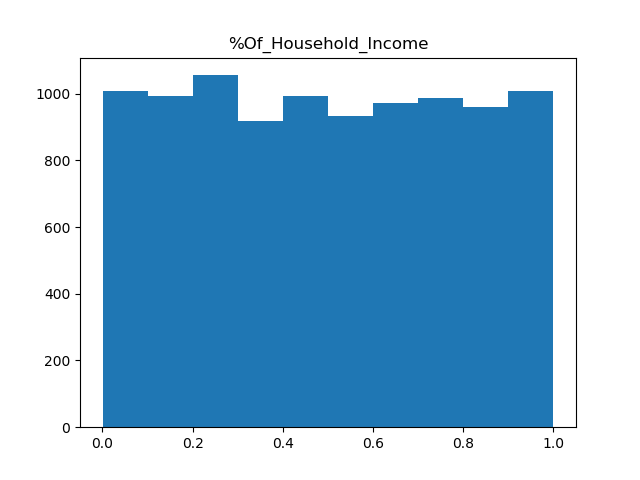
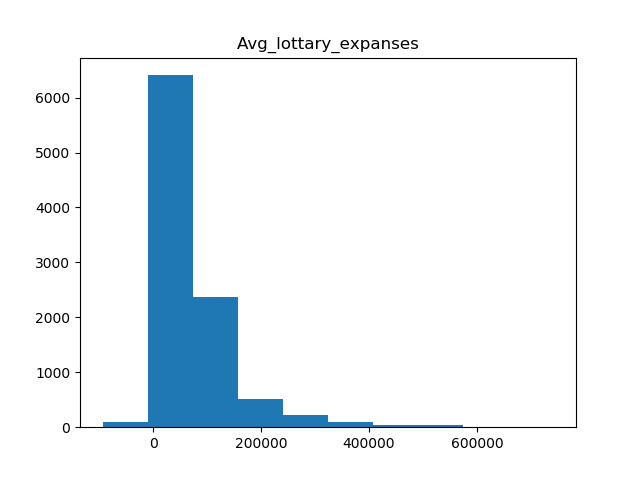
**In this part we'll describe the actions that we made on the data given to us in order to prepare it for future leaning algorithm.**

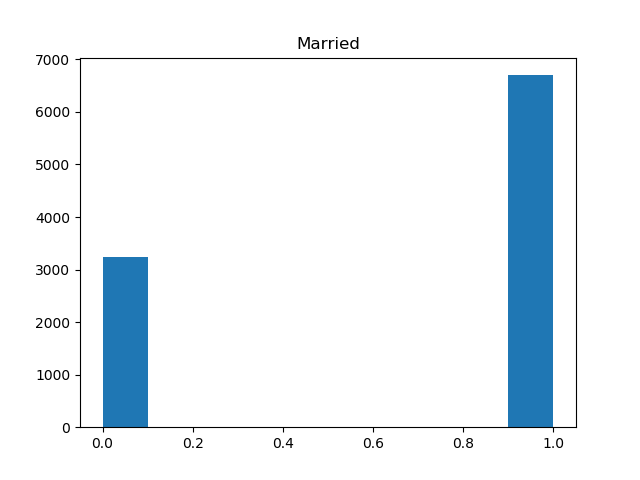
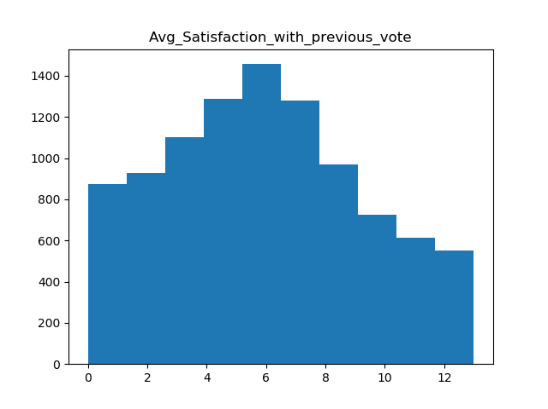
**First we'll describe the process that made us understand better the data were working on:**

1. **After we loaded the data we separated the features into two types: numerical features and nominal features**

|  |  |
| --- | --- |
| **Nominal Features** | **Numerical Features** |
| Vote | Occupation\_Satisfaction |
| Age\_group | Avg\_monthly\_expense\_when\_under\_age\_21 |
| Looking\_at\_poles\_results | Avg\_lottary\_expanses |
| Married | Avg\_monthly\_expense\_on\_pets\_or\_plants |
| Gender | Avg\_environmental\_importance |
| Voting\_Time | Financial\_balance\_score\_(0-1) |
| Will\_vote\_only\_large\_party | %Of\_Household\_Income |
| Most\_Important\_Issue | Yearly\_IncomeK |
| Main\_transportation | Avg\_size\_per\_room |
| Occupation | Garden\_sqr\_meter\_per\_person\_in\_residancy\_area |
| Financial\_agenda\_matters | Avg\_Residancy\_Altitude |
|  | Yearly\_ExpensesK |
|  | %Time\_invested\_in\_work |
|  | Avg\_education\_importance |
|  | Avg\_Satisfaction\_with\_previous\_vote |
|  | Avg\_monthly\_household\_cost |
|  | Phone\_minutes\_10\_years |
|  | Avg\_government\_satisfaction |
|  | Weighted\_education\_rank |
|  | %\_satisfaction\_financial\_policy |
|  | Avg\_monthly\_income\_all\_years |
|  | Last\_school\_grades |
|  | Number\_of\_differnt\_parties\_voted\_for |
|  | Political\_interest\_Total\_Score |
|  | Number\_of\_valued\_Kneset\_members |
|  | Num\_of\_kids\_born\_last\_10\_years |
|  | Overall\_happiness\_score |

1. In order to handle the data with more accuracy, we looked at the distribution of each of the features by making histograms of the values of each feature:





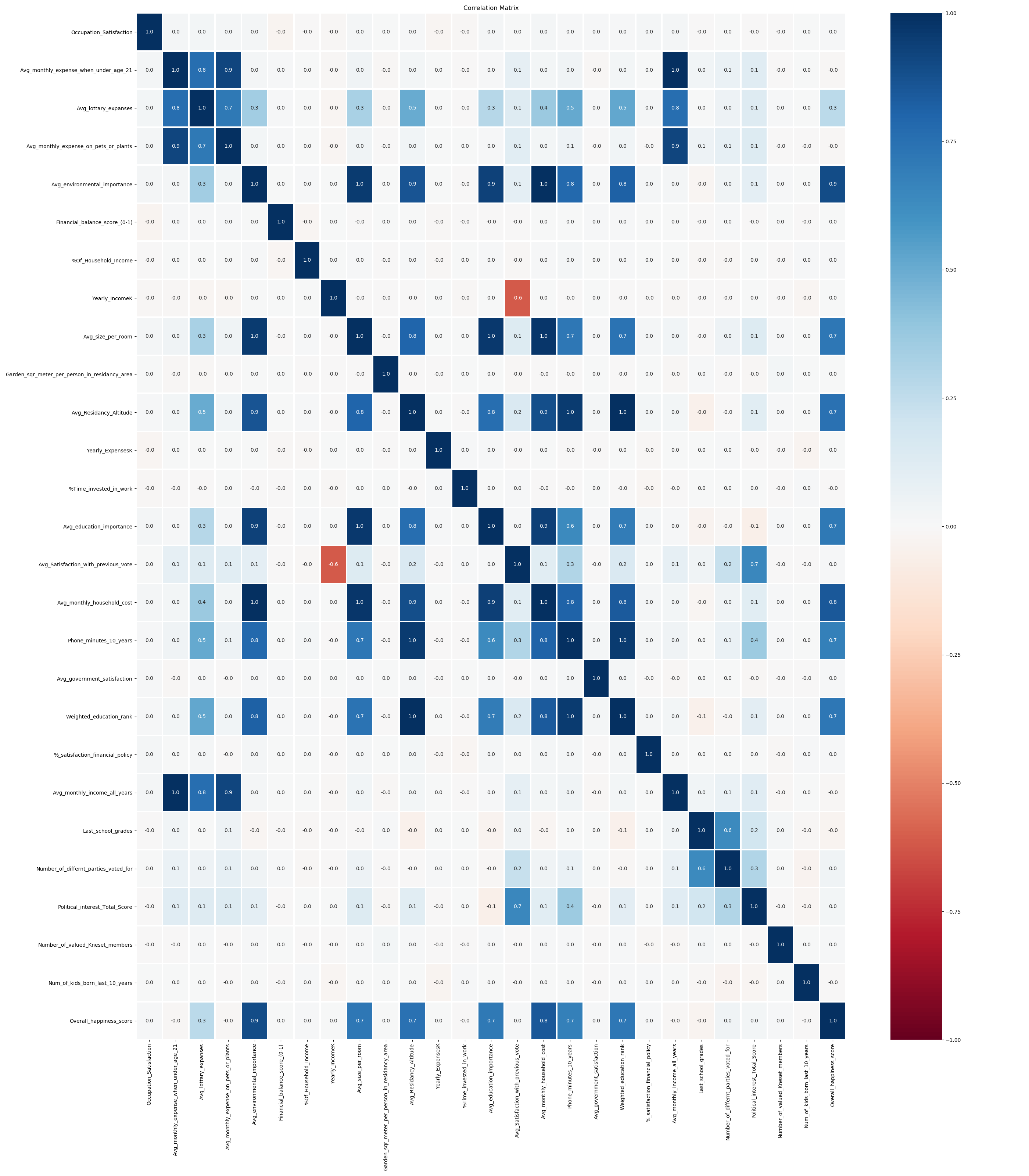
We can learn from the graph how the features are distibuted and which of the features are distibuted uniformly or normally.

1. We checked to see if there is a linear correlation between the features in case we need to fill missing data. We used a heatmap (supplied by the seaborn library) and generated a plot that showed for each pair of features their correlation:

Blue – positive correlation

Red – negative correlation

As the Absolute value of the value becomes greater so does the color gets darker.



1. From the names of the features we understood that their values must be non-negative.

**The algorithm of the data preparation:**

1. Load ElectionsData.CSV
2. Identify nominal features and map their values into integers
3. Split the data file into 3 separate files: train, validation and test for further changes.
4. Remove all negative values and change them to Nan
5. For every feature the distributed normally do standardization according the Z threshold and swich every value above 4.5 with Nan (needed because of the great gap between some values and the distributed valuses of some features)/
6. Complete the missing values:
   1. For the features that had a high correlation (above 0.9) with other features we calculated the missing values according to effective linear coefficient between the two features.
   2. We used the closest fit algorithm for each set of data and filled missing values by the closest values.
   3. For each set we executed expectation maximization algorithm
   4. For the lasting remaining nominal values we completed the missing values to be the common value
   5. For the lasting remaining numeric value we set the values to the average of the feature.
7. Scaling:
   1. For features that distributed uniformly we scale them to the range [-1,1]
   2. For features that distributed normally we standardized according to Z
   3. Nominal features remained the same.
8. Feature selection:
   1. Filter method: for the numeric features we removed the features that their variance was small than the threshold 0.2 since those features didn’t contributed more information therefore can be removed. The features that were removed were:

* Avg\_education\_importance
* Avg\_environmental\_importance
* Avg\_Residancy\_Altitude
  1. Wrapper method: By using SGD-Classifier and selectKBest from the sklearn library we were able to found the group of features with the atmost mutual information and with the size of 16 features which benefit that information.

The chosen features are:

* Avg\_monthly\_expense\_when\_under\_age\_21
* Avg\_lottary\_expanses
* Avg\_Satisfaction\_with\_previous\_vote
* Avg\_education\_importance
* Avg\_Residancy\_Altitude
* %Time\_invested\_in\_work
* Avg\_monthly\_expense\_on\_pets\_or\_plants
* Avg\_monthly\_household\_cost
* Phone\_minutes\_10\_years
* Avg\_size\_per\_room
* Weighted\_education\_rank
* Avg\_monthly\_income\_all\_years
* Last\_school\_grades
* Political\_interest\_Total\_Score
* Number\_of\_valued\_Kneset\_members
* Married
* Most\_Important\_Issue

1. Saved the changed data sets (3) to a new csv files with their selected feature and fixed values.

**First Bonus – Relief Algorithm:**

1. To identify the role of the attributes with respect to the Vote label, first we mapped the values of the vote label into numeric values:

'Blues'**:** 0**,**

'Browns'**:** 1**,**

'Greens'**:** 2**,**

'Greys'**:** 3**,**

'Khakis'**:** 4**,**

'Oranges'**:** 5**,**

'Pinks'**:** 6**,**

'Purples'**:** 7**,**

'Reds'**:** 8**,**

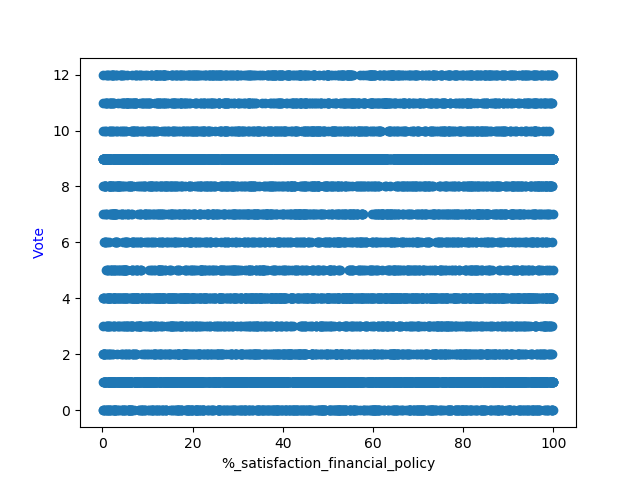
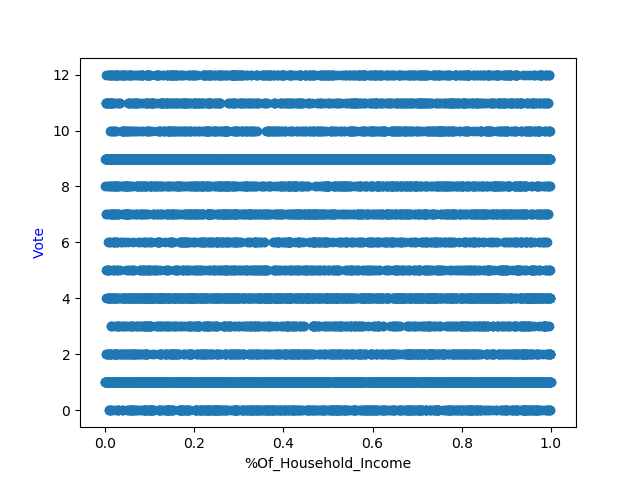
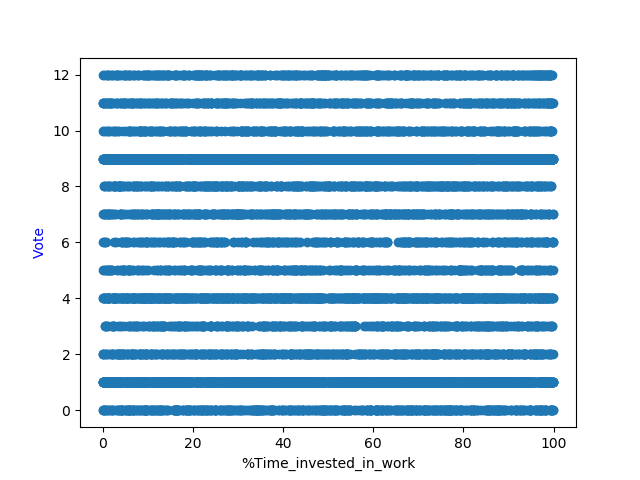
'Turquoises'**:** 9**,**

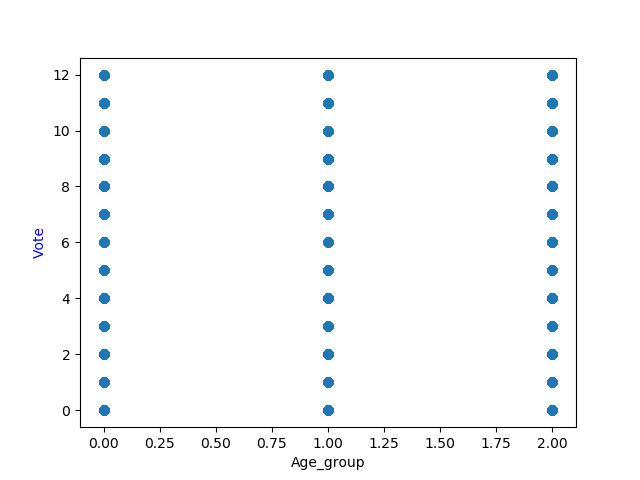
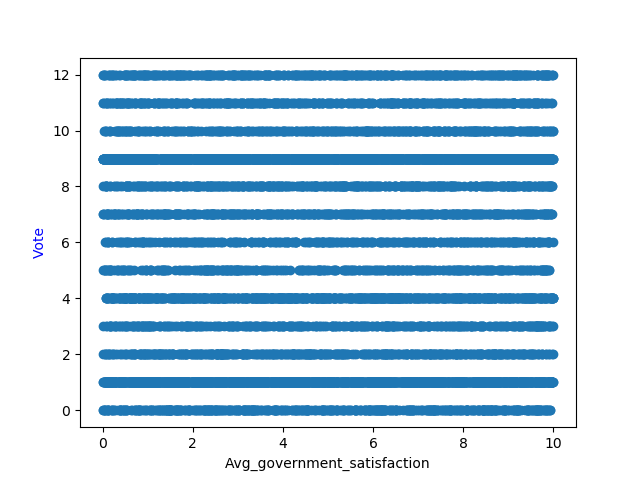
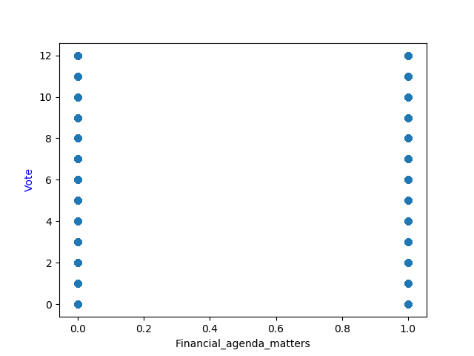
'Violets'**:** 10**,**

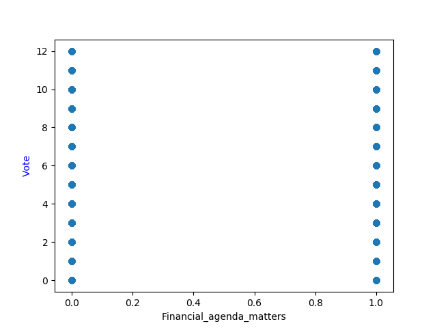
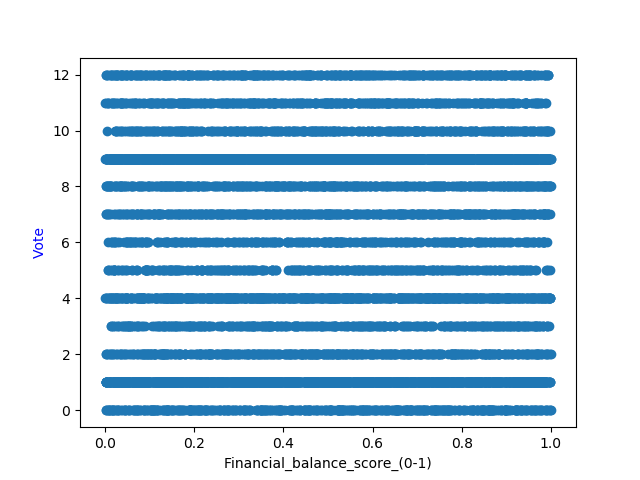
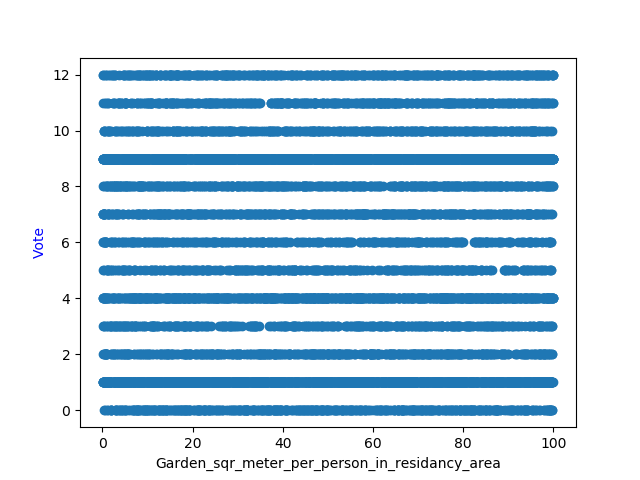
'Whites'**:** 11**,**

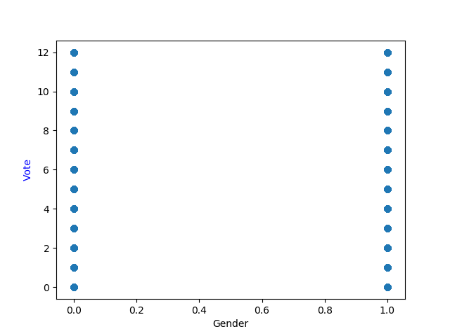
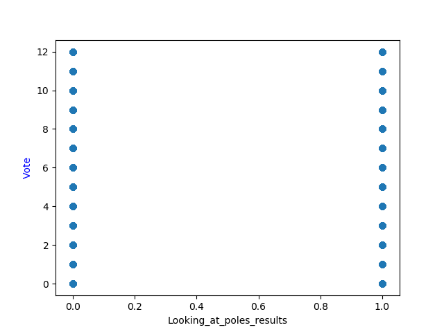
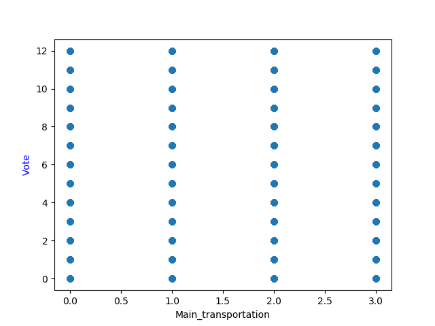
'Yellows'**:** 12

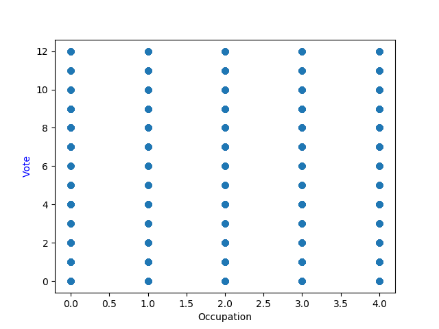
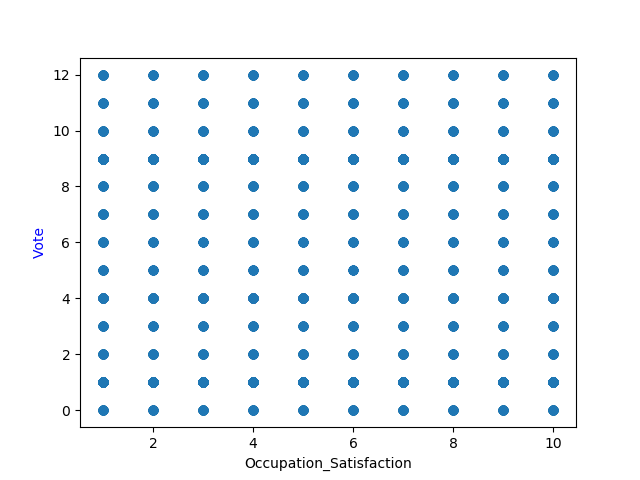
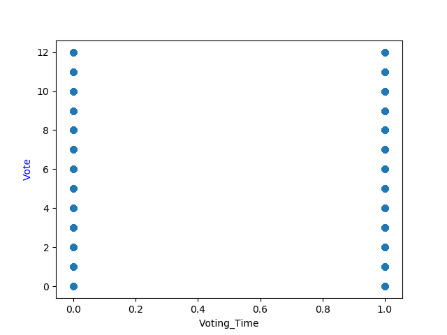
Features we couldn't Conclude a connection between them to the vote label are those features that their values distributed uniformly over the vote label:

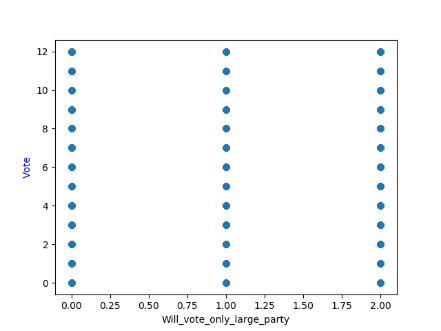
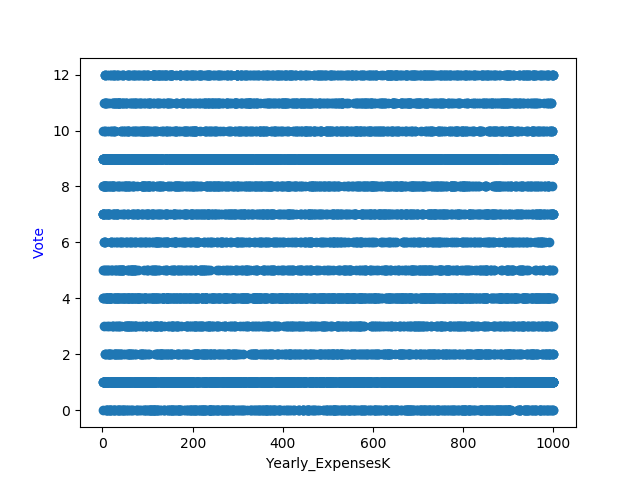
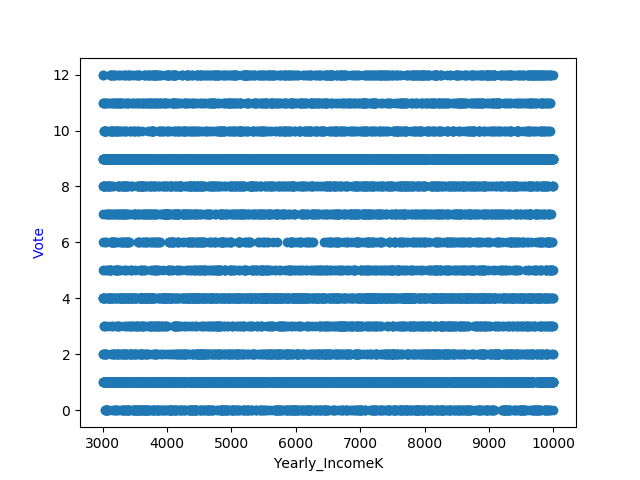
  

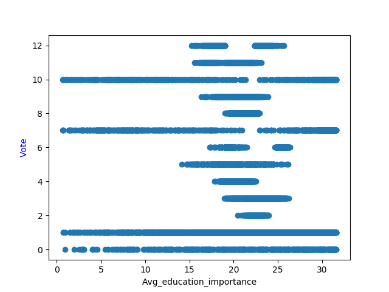
  

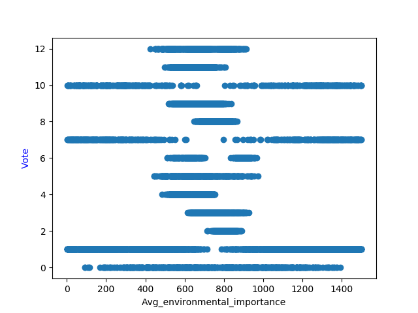
  

For these features we found a connection to the vote label:

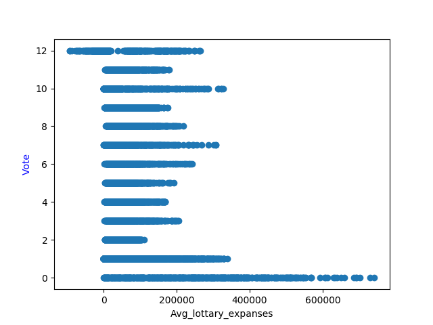
* Avg\_education\_importance: the Blue, Brown,Purpule and Violets are common among the entire range while the other colors are common only among the avg values



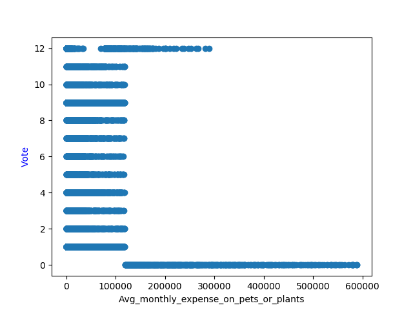
* Avg\_environmental\_importance: the Blue, Brown,Purpule and Violet are common among the entire range while the other colors are common only among the avg values



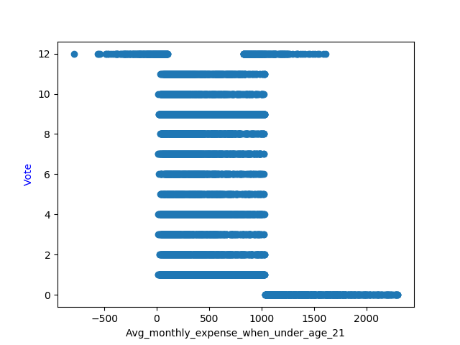
* Avg\_lottary\_expanses: people with high expanses tend to vote blue while people with low expanse tend to vote yellow.



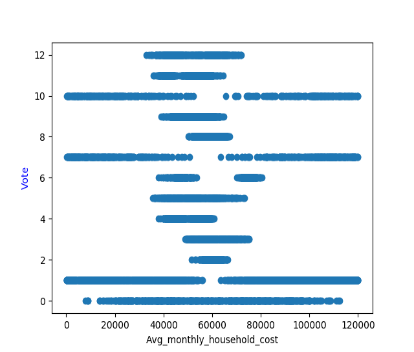
* Avg\_monthly\_expense\_on\_pets\_or\_plants: people with high expanses tend to vote blue while people with low expanse tend to vote yellow.



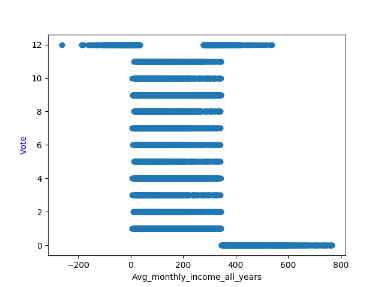
* Avg\_monthly\_expense\_when\_under\_age\_21: people with high expanses tend to vote blue while people with low expanse tend to vote yellow.



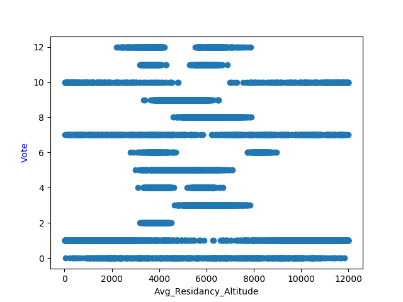
* Avg\_mothly\_household\_cost: the Blue, Brown,Purpule and Violet are common among the entire range while the other colors are common only among the avg values



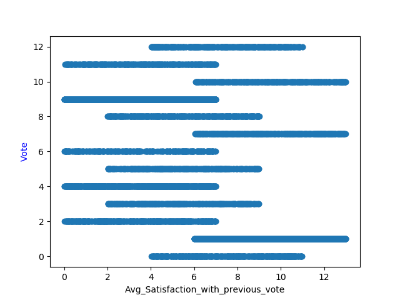
* Avg\_monthly\_income\_all\_years: people with high expanses tend to vote blue while people with low expanse tend to vote yellow.



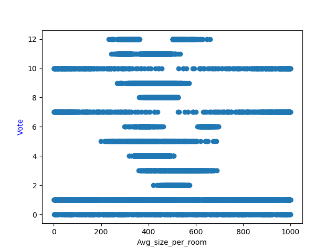
* Avg\_Residancy\_Altitude: the Blue, Brown,Purpule and Violet are common among the entire range while the other colors are common only among the avg values.



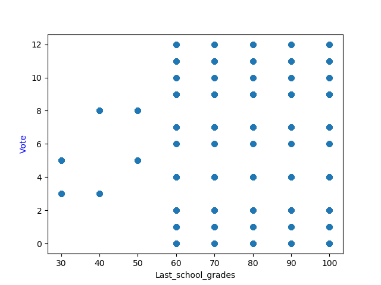
* Avg\_Satisfation\_with\_previus\_vote: for each group of satisfaction levels there is a different types of colors they choose.



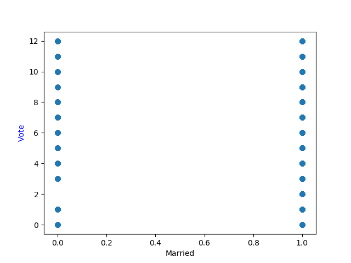
* Avg\_Size\_Per\_Room: the Blue, Brown,Purpule and Violet are common among the entire range while the other colors are common only among the avg values.



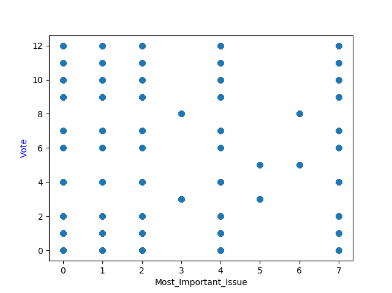
* Last\_School\_grades: people with low school grades tend to vote to grey, orange and red.



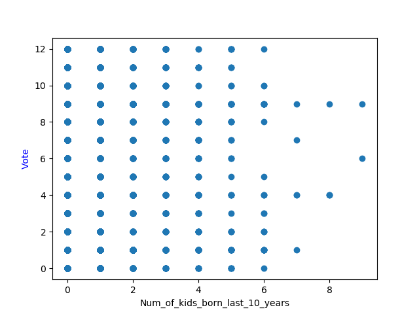
* Married: single people tend to not chose green.



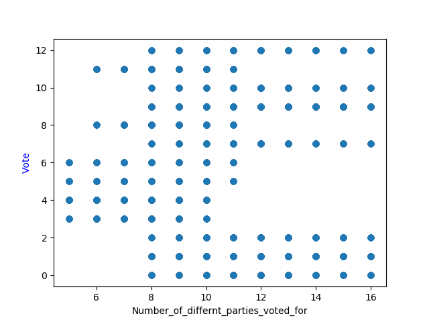
* Most\_Important\_Issue: the red, orange and grey are common among specific values.



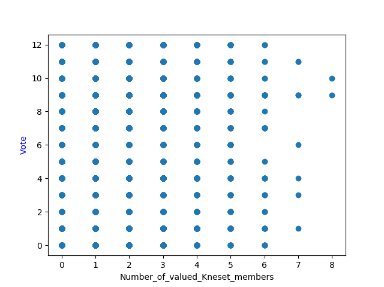
* Num\_of\_kids\_born\_last\_10\_years: families with high number of small children tend to chose brown, khaki, pink' purple and turquoises.



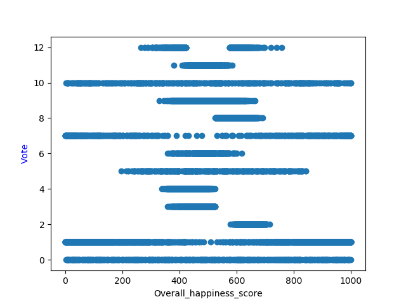
* Number\_of\_diferrent\_parties\_voted\_for: mixed trend among high and low values.



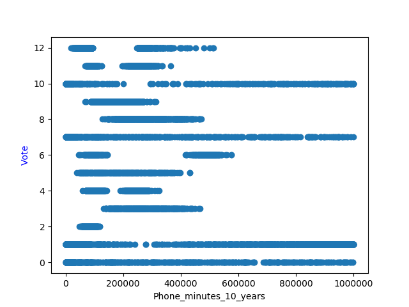
* Number\_of\_valued\_kenset\_members: the high values tend to choose specific colors such as: Turuoise pink and Khaki.



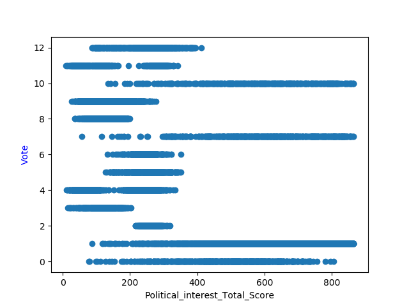
* Overall\_hapiness\_score: the Blue, Brown,Purpule and Violet are common among the entire range while the other colors are common only among the avg values.



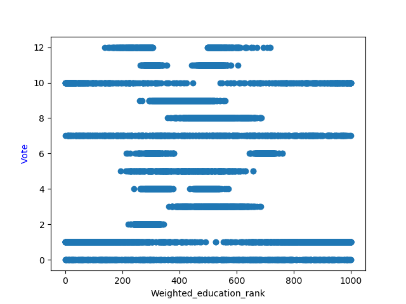
* Phone\_minutes\_10\_years: the Blue, Brown,Purpule and Violet are common among the entire range while the other colors are common only among the lower values.



* Political\_interest\_total\_score: the Blue, Brown,Purpule and Violet are common among the entire range while the other colors are common only among the lower values.



* Weighted\_education\_rank: the Blue, Brown,Purpule and Violet are common among the entire range while the other colors are common only among the avg values.



1. The relief algorythm is located at relief.py. The function gets the following parameters:

• x\_train: train data frame

• y\_train: train labels data frame

• nominal\_features\_list: list of the nominal features

• numerical\_features\_list: list of the numerical features

• 𝑁 , num\_of\_iterations: number of iterations

• 𝜏 , threshold: threshold

After n iterations the function relief return the selected features.

For 𝜏=1 and N=1000 We got these selected features:

* 'Occupation\_Satisfaction'
* 'Avg\_lottary\_expanses'
* 'Avg\_monthly\_expense\_on\_pets\_or\_plants'
* 'Gender', 'Avg\_Residancy\_Altitude'
* 'Yearly\_ExpensesK', 'Avg\_government\_satisfaction'
* 'Avg\_monthly\_income\_all\_years'
* 'Political\_interest\_Total\_Score'
* 'Number\_of\_valued\_Kneset\_members'
* 'Financial\_agenda\_matters'
* 'Num\_of\_kids\_born\_last\_10\_years'

**Feature selection capabilities of Relief vs the other methods that we have used**

Pros:

* Feature selection Can handle large amount of samples
* Features selection can be controlled by setting a threshold
* Using statistical analysis **only**
* Since we don't need to search after the best sub set of the feature and nor do we scale the set of the chosen features the algorithm run at polynomial time.
* Easy to implement
* not affected from feature dependency

Cons:

* Not always clear what is the best threshold for optimal results
* Affected from calculations that based on samples distance, for example- every feature is given an equal weight when the distance is calculated (nominal feature that translated to integer value can change the distance although the distance between their values is insignificant)
* Need many iterations in order to get clear results.

Second Bonus – SFS Algorithm:

1. The algorithm was implemented in the file sfs.py in the function sfs\_algo. It's parameters:

* x\_train: train data frame
* y\_train: train labels data frame
* clf: classifier to examine
* subset\_size: user required subset size not mandatory

The algorithm is implemented using scoring of a Kcross fold validation (k=3) and returning the selected features. The selected geatures set is determined by the subset\_size if some of the features improve the classification.

We examined the algorithm with two different classifiers:

* SGDClassifier:

SVM Classifier accuracy score before SFS is: 0.7635317926158812

SVM Classifier selected features are: ['Political\_interest\_Total\_Score', 'Avg\_monthly\_expense\_on\_pets\_or\_plants', 'Last\_school\_grades', 'Number\_of\_differnt\_parties\_voted\_for', 'Avg\_monthly\_household\_cost', 'Avg\_environmental\_importance', 'Phone\_minutes\_10\_years', 'Overall\_happiness\_score', 'Married', 'Avg\_Satisfaction\_with\_previous\_vote', 'Yearly\_IncomeK', 'Avg\_lottary\_expanses', '%Time\_invested\_in\_work', 'Avg\_education\_importance']

SVM Classifier score after SFS is: 0.795706953396898

* KNN:

K Neighbors Classifier score before SFS is: 0.6151617071259761

K Neighbors Classifier selected features are: ['Weighted\_education\_rank', 'Avg\_size\_per\_room', 'Overall\_happiness\_score', 'Last\_school\_grades', 'Number\_of\_differnt\_parties\_voted\_for', 'Phone\_minutes\_10\_years', 'Avg\_monthly\_household\_cost', 'Political\_interest\_Total\_Score', 'Avg\_education\_importance', 'Avg\_Residancy\_Altitude']

K Neighbors Classifier score after SFS is: 0.8471659890727626

* 1. **feature selection capabilities of SFS vs the other methods that we have used:**

Pros:

* Can deal with large amount of samples
* Can control the size of the selected features
* the selected features are determined by measures that can be changed by the user
* using classification algorithm we can create a learning curve that help us learn about the problem
* Easy to implement

Cons:

* We can't undo and reselect a feature even if it's completely redundant.
* Since the features are selected by the accuracy to the training set, there is a chance that there will be an over-fitting of the selected features.
* Algorithm run time depends on the number of features so it's preferable to run it on a small number of features.
* The algorithm chose to include a feature only if that feature improve the prediction accuracy and therefore doesn't take into account feature dependencies.
* The selecting features methodology is incremental thus it may not check other possibilities of other sub-sets of features that may have even higher accuracy level.