Date: December 3rd, 2019
1030: Hands on Data-Science
Instructor: Andras Zsom
Advisor: Joachim Krueger
Student: David Kebudi
GitHub Repository: https://github.com/Kebudi/Data1030-Final-Project
Data Source: Prelec, MIT

Predicting Binary Outcomes using Wisdom of Crowds

Brown University Data Science Department

1. Introduction

1.1 The Problem

In 2017, Professor Prelec of MIT Psychology published a **study** on predicting the outcome of binary questions using Bayesian modeling. In his paper he suggested that if we have data on: the question, participant's answer, participant's guess on how many people will say that the question's answer is **TRUE**, and participant's confidence in his own answers, we would be able to build a model that predicts the answer of the question accurately. He reached an accuracy of 73.3% with his model.

The main goal of the project is to come up with a model that uses the same inputs as Prelec, but classifies the true reposes to the question with better accuracy. **The target variable for the project is the true answer, which is in the binary form: 1/0.**

1.2 The Data

The total number of participants across the 3 datasets is 97, with 7 features for each one of them in every dataset = [‘**binary_question**, '**topic**', '**subject**', '**own**', '**actual**', '**meta**', '**confidence**’]. This data is provided by Professor Prelec of MIT, which he collected through studies at Princeton and MIT.

The data sets come in 3 pieces, with each one on a different topic:

(1) Binary questions on the capitals of states in the USA surveyed at MIT [*example question: Albany is the capital of New York State.*]

(2) Binary trivia questions on many different subject surveyed at Princeton University, [*example question: George Bush Jr. is the youngest president in the US history.]*

(3) Binary question on malicious moles on the body asked to dermatologist [*example question [in picture format]: This mole is malicious.]*

1.2.1 Variables

PRELEC FEATURES:

- **Binary Question:** the question asked to the participants, which the participant evaluates the truth of.
- **Topic:** the topic of the question This is a categorical variable — dataset_1: (‘*state*’), dataset_3: (‘*dermatology*), and 80 different values in dataset_2, depending on the category of the trivia question.
- **Subject:** the unique ID for the participant
- **Own:** the response of the participant to the binary question (1/0)
- **Actual**: the correct response to the binary question (1/0)
- **Meta:** the continuous variable between 0-100, the percentage of the study group the

participant think will vote TRUE

- **Confidence:** the continuous variable between 50-100, the percentage of confidence the participant has for the response he has given.

As a first step, we need to have the meta and confidence continuous variables to be bucketed into categories. This way we would be able to normalize the different datasets into a comparable form by dividing the number of people who fall into the bucket by the number of unique people in the dataset.

ENGINEERED FEATURES:
Other than the features given by Prelec, I also engineered 7 sub-categories of features.

- **Self Consensus Score (sc):** given that meta is the guessed percentage of people voting TRUE, people who are voting TRUE are guessing the percentage of the population that will agree with them. Thus, we calculate self consensus by:

$$if\ \ own = 1 \rightarrow sc = meta$$

$$if\ \ own = 0 \rightarrow sc = 1 - meta$$

Self consensus score is bucketed like meta and confidence.

- N**umber of participants**: the number of participants in the question group.

- **Own Percentage**: This is the percentage of people who answered TRUE to the question.

- **Minority Percentage:** The percentage of the people in the dataset that gave the minority response.

- **Skew:** The horizontal and vertical skews of the distributions of the variables. Vertical Skew: confidence, sc, and meta scores. Horizontal Skew is for each question's own skew of its continuous variables.

- **Description [std, mean, min, max, percentiles]:** This is the distribution and statistical values of confidence, sc, and meta for each question.

- **Majority Prediction:** This feature indicates what the prediction for a majority accepting model would be. [*1 if more than 50% of the population voted 1*]

- **Prelec Prediction:** for each question, this feature indicates what the prediction for Prelec's model would be

- **Predicted number of experts in the data sample:** This variable is a continuous variable of the number of experts in the data. The definition of an expert is as follows:

$$in\ minority \cap correct$$

st. if you guess the question right and you are in the minority, you are an expert.

Then, I created a new variable: c-sc, c = confidence. c-sc should be high for experts, and low for non-experts, since we expect the expert to be confident in his answer, and have low consensus with himself. Then I conducted a t-test to prove that the differences in the means of c-sc for experts and non-experts was significant.

As a result I calculated a static variable, the Krueger-stat:

$$K = \frac{\mu(c-sc)_{experts} - \mu(c-sc)_{non-experts}}{\sigma}$$

For datasets where expert number = 0:

$$\hat{K} = \frac{0 - \mu(c-sc)_{non-experts}}{\sigma}$$

Then for each question with a minority, I ran the following code (in pseudo-code):

```
for i in range(len(minority):
        i = i+1

        # then calculate k-stat
        for the given minority
        cutoff st. i number of
        people in the minority
is      considered to be  an
        expert

        k = calculate_k_stat(i)

        compare(k, K)
```

where K and $\hat{K}$ are the k-stats calculated in the training data.

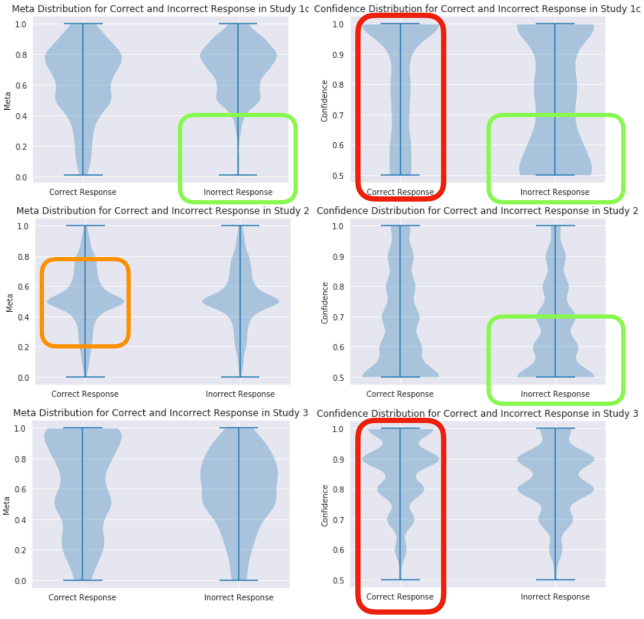
The compare function decides which *i* bring the data closest to either of the K values. If *i* brings it closer to $\hat{K}$, then the predicted expert number is 0, else, whatever *i* is, when it is closest to K.

i is the predicted number of experts.

Overall, the data's dimensions before preprocessing is **210x62.**

2. EDA

2.1 Meta and Confidence Distributions

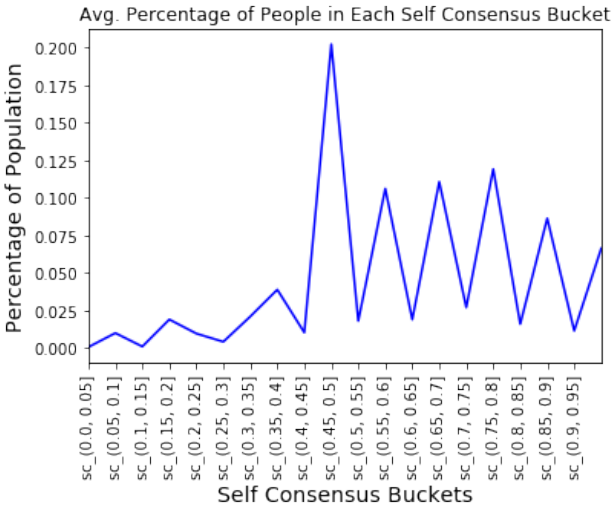


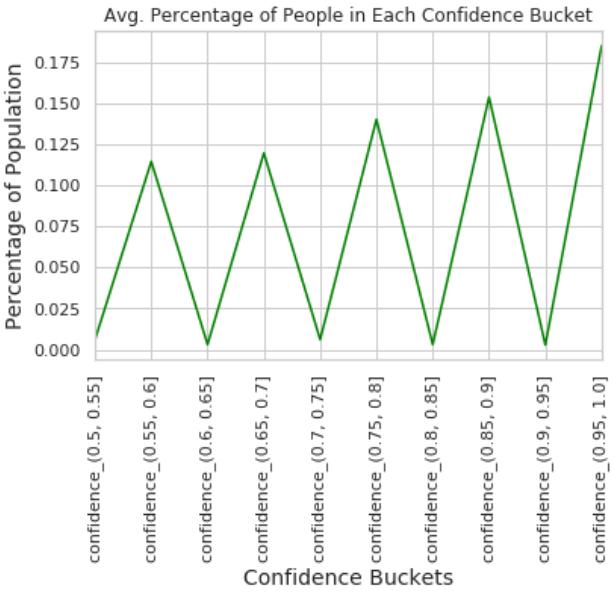
GREEN: people giving the incorrect response tend to have a more evenly distributed confidence value across.

RED: people with the correct response tend to have higher confidence.

ORANGE: people who have correctly responded to the question, also are more inclined to say FALSE if they believe so, given the bulking around 0.5.

2.2 Average Percentage of People in Each Bucket





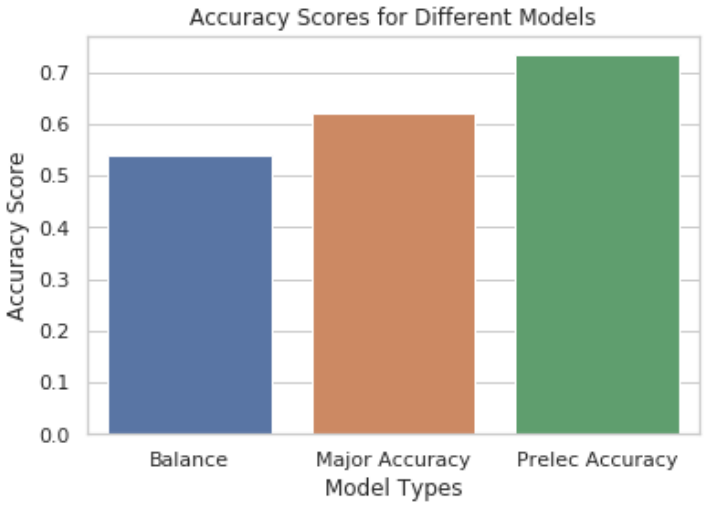
There graphs show that 20% of the people believe that at least 50% of the rest will agree with their response. More people are more confident.

2.4 Accuracy Scores

There are three main base accuracies my model is competing against. (1) Balance, the balance of the data: 58%. (2) The majority accuracy is the classification technique Prelec competes against: 64%. Prelec's model: 73.3%. Prelec follows the following classification algorithm:

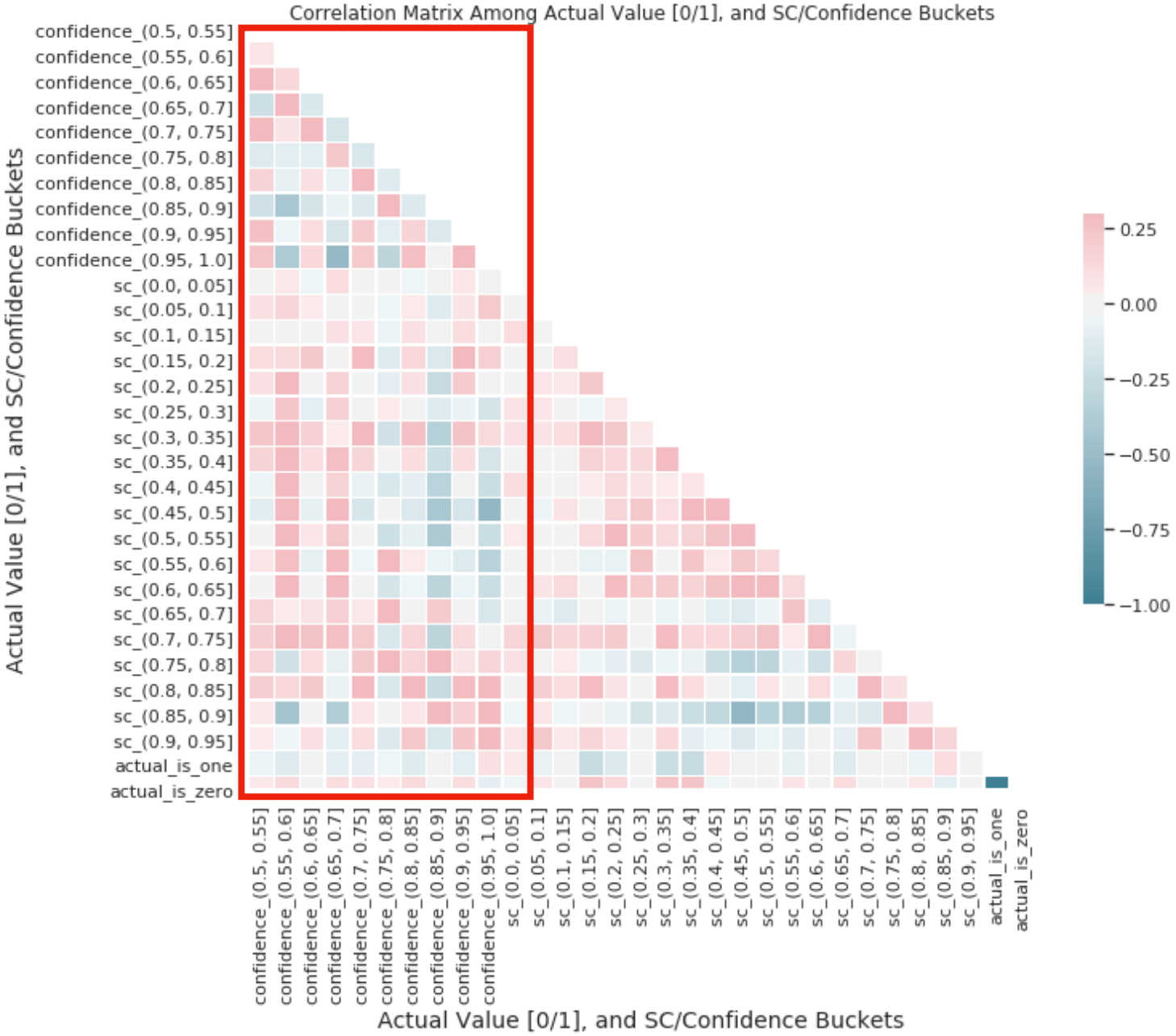
```
if average meta per question
> percentage of people
saying 1:

     Prediction = 0
else:
     Prediction = 1
```



2.3 Correlation Matrix

We observe: **(1)** if the actual response of the question is TRUE, then it has a positive correlation with the confidence values. If it is FALSE, it has a negative correlation.


Correlation Matrix Among Actual Value [0/1], and SC/Confidence Buckets
Actual Value [0/1], and SC/Confidence Buckets
confidence_(0.5, 0.55]
confidence_(0.55, 0.6]
confidence_(0.6, 0.65]
confidence_(0.65, 0.7]
confidence_(0.7, 0.75]
confidence_(0.75, 0.8]
confidence_(0.8, 0.85]
confidence_(0.85, 0.9]
confidence_(0.9, 0.95]
confidence_(0.95, 1.0]
sc_(0.0, 0.05]
sc_(0.05, 0.1]
sc_(0.1, 0.15]
sc_(0.15, 0.2]
sc_(0.2, 0.25]
sc_(0.25, 0.3]
sc_(0.3, 0.35]
sc_(0.35, 0.4]
sc_(0.4, 0.45]
sc_(0.45, 0.5]
sc_(0.5, 0.55]
sc_(0.55, 0.6]
sc_(0.6, 0.65]
sc_(0.65, 0.7]
sc_(0.7, 0.75]
sc_(0.75, 0.8]
sc_(0.8, 0.85]
sc_(0.85, 0.9]
sc_(0.9, 0.95]
actual_is_one
actual_is_zero
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Actual Value [0/1], and SC/Confidence Buckets

3. Methods

3.1 Preprocessing

- **Binary Question:** no encoder, it is the index of the dataset
- **Topic:** OneHotEncoder, this will be used to understand if topic of a certain type have more impact in the model or not. Topics are given as strings and cannot be ordered.
- **Own:** no pre-processing, it is binary [0,1]
- **Actual**: TARGET VARIABLE, Label encoder is used.
- **SC:** no pre-processing, the data is continuous between [0-1] for all 20 buckets
- **Confidence:** no pre-processing, the data is continuous between [0-1] for all 10 buckets
- **Number of people:** StandardScaler, because is is an integer with no maximum value.
- **[all] Description [std, mean, min, max, percentiles]:** no preprocessing, values are between 0 and 1 for confidence, meta, and sc
- **[all] Skew:** Standard Encoder because skew can go as low as below zero, and go higher than +1. There is no min and max.
- **Predicted number of experts in the data sample:** StandardScaler, because is is an integer with no maximum value (only internally — can't be more than population size)
- **Prelec/Majority Predictions:** no preprocessing, binary variables
- **Minority Percentage:** distributed between 0 and 1. No preprocessing.
- **Own Percentage:** percentage of people voting TRUE, always between 1 and 0. Thus, no preprocessing.

After preprocessing, I have the shape of the data is (210, 141).

3.2 Models

There are 4 classifiers I used in my supervised machine learning pipeline: Nearest Neighbor, Logistic Regression, Random Forrest, and Super Vector Machine (SVM). In order to do cross-validation (CV), and thus not over-fit the model, I used KFoldSplitting(n=5). This allowed me to not have any data leakage either, and train my model with 60% of the data, cross validating with 20% and then running it on 20% test data. Given that my data is iid *(independently and identically distributed)*, normal KFold method performed successfully.

3.2.1 Hyper Parameters

- **Nearest Neigbour**
  1. **n_neighbors:** 20 integers, randomly selected between 2 and 100
  2. **weights:** distance and uniform, giving each neighbor a different

weight based on it's distance or not (equally distributed)
3. **metric:** euclidean, manhattan, *the metrics used to measure distance*

- **Logistic Regression**
  4. **C:** 100 floats between 10^{-5} and 10^{4}
  5. **penalty:** "lasso" or "ridge"

- **Random Forrest**
  6. **max_features:** 'auto', 'sqrt', 'log2'
  7. **max_depth:** 10 integers, randomly selected between 2 and 100, *limiting over fitting by not giving a very large number*
  8. **min_samples_split:** 10 integers, randomly selected between 2 and 100

- **SVM**
  9. **C:** 30 floats between 10^{-4} and 10^{4}
  10. **gamma:** 30 floats between 10^{-4} and 10^{4}

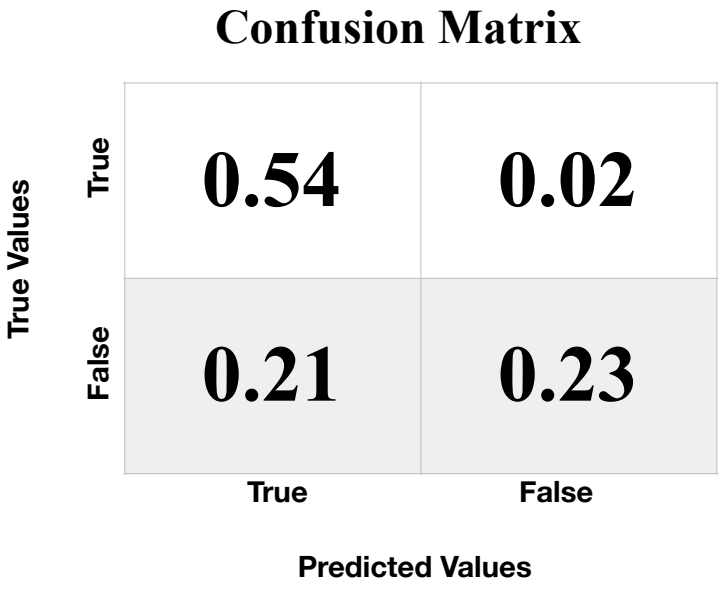
3.2.2 Model Table

In order to account for the indeterministic nature of the methods, I passed in 9 different random seeds to the pipelines. The table below has the best accuracy of the 9 passes, and the mean accuracy of the passes with its standard deviation.

	Best Accuracy (Random Seed)	Average Accuracy	Standard Deviation of Accuracy
Logistic	**0.75 (1090)**	**0.7156**	**0.0226**
SVM	**0.7738 (436)**	**0.7162**	**0.0338**
Random Forrest	**0.8154 (872)**	**0.7625**	**0.0381**
Nearest N.	**0.7619 (654)**	**0.7109**	**0.0294**

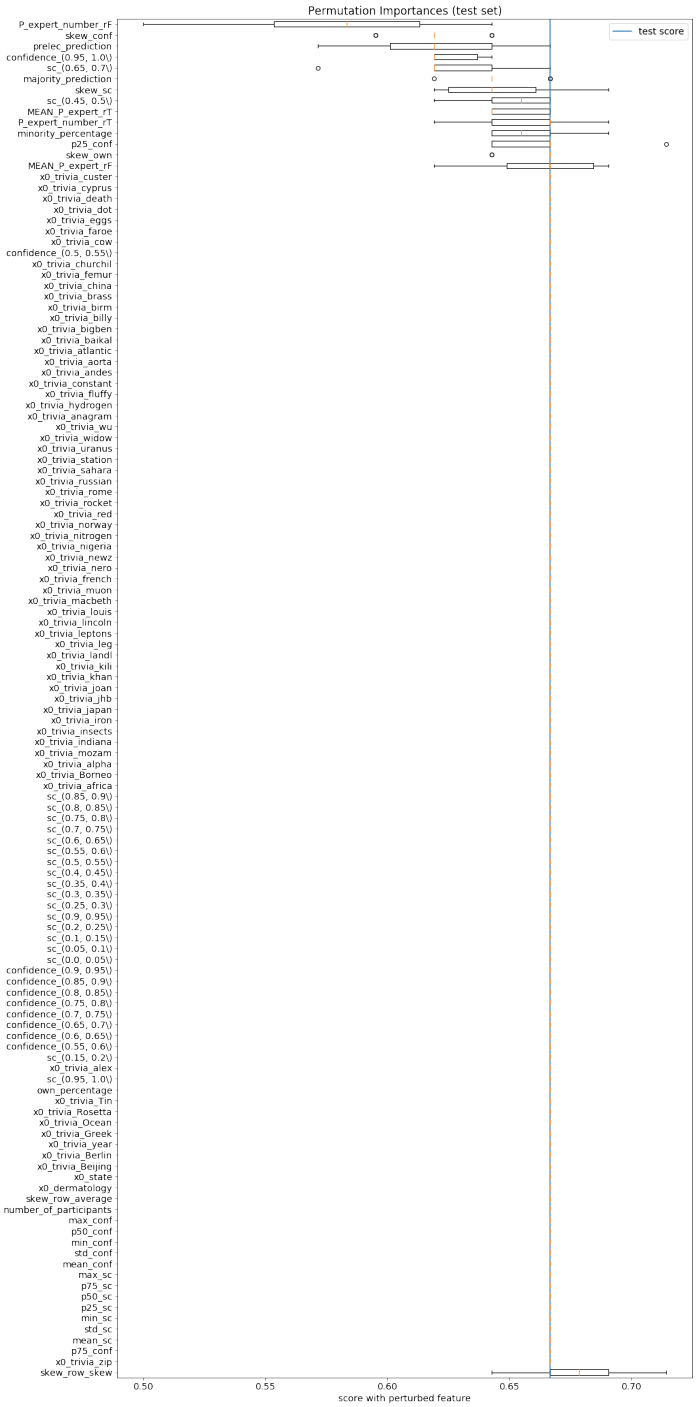
I used the accuracy metric = *"accuracy score",* because the problem is a simple binary classification problem. According to my accuracy score, Random Forrest is the most accurate model for this problem.

4. Results



We see on the confusion matrix that the best model, with the best accuracy, Random Forrest with random_seed=872, has 44 false positives, which is 21% of the

dataset. The accuracy of the best performing model, 0.81, is significantly higher than both our balance = 0.53 and Prelec's prediction accuracy score = 0.73.



The most important 6 features according to the global feature selection are (in the order of importance):

1. **p_expert_number_rF**
2. **skew_conf**
3. **prelec_prediction**
4. **conf(0.95,1)**
5. **sc(0.65,7)**
6. **majority_prediction**

This result is not surprising. Prelec_Prediction, due to it's high accuracy, and majority_Prediction due to it's predictive nature. It is interesting to see how the predictions of expert numbers play a significant part in the model.
The parameters for the best Random Forrest Model are:

- **max_depth=11**
- **max_features='auto'**
- **min_samples_leaf=1**
- **min_samples_split=7**
- **n_estimators=100**
- **random_state=872**

Using this model, we will be able to classify binary questions after surveying an acceptable population. This theory, while not new, is an innovation in the Wisdom of Crowds calculations. What is the new is the use of Machine Learning and the resulting increase in accuracy

5. Outlook

There are 3 ways through which this model can be further advanced:

1. **XGBoost Classifier**

The model can greatly benefit from a more robust, and accurate machine learning model. The tree based algorithm, will likely return better resultss.

2. **Data about Unknown Future Guesses**

While the model promises to leverage the Wisdom of Crowds social theory for making predictions, the data at hand only allows us to predict the outcome of statements which cannot be fact checked. Gathering data about future guesses such as: *Markets will go up next week,* would improve accuracy.

3. **Expert Classification Model**

It would be really interesting to see how a model that goes through the initial datasets and classifies each participant as an expert performs for each question. Since our definition of expert inherently bases itself on giving the true response:

$$in\ minority \cap correct$$

The response of the expert classified person would be the prediction for the question.

6. References

Papers:

Prelec, Dražen, et al. “A Solution to the Single-Question Crowd Wisdom Problem.” *Nature*, vol. 541, no. 7638, 2017, pp. 532–535., doi:10.1038/nature21054.

Data:

The data was provided by Professor Dražen Prelec of MIT.