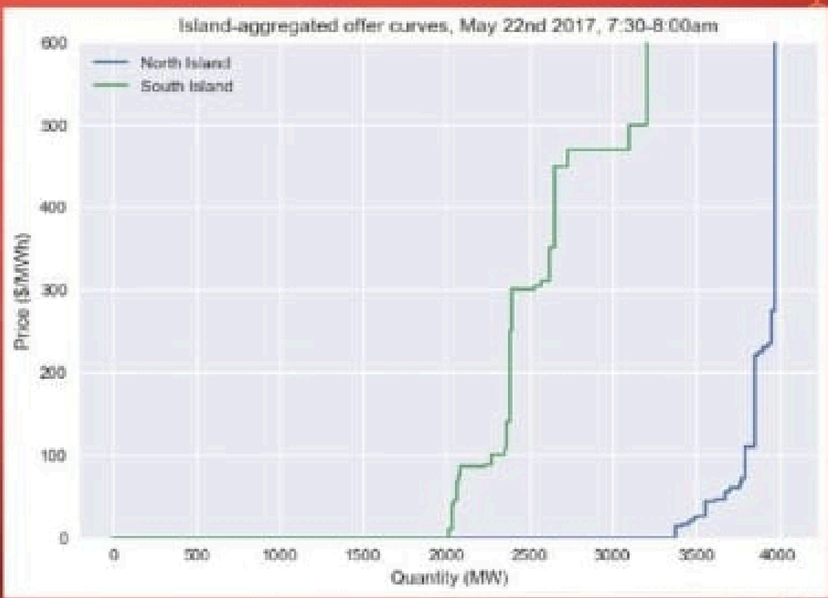

Forecasting Electricity Price
in International Electricity
Market Using Neural Networks

AIMS

- Use machine learning to produce accurate predictions of demand and nodal prices two hours in advance.
- Base price predictions on aggregated North and South Island offer stacks as per WITS.



AGGREGATE MARKET CLEARING MODEL

- Could solve a linear program with two nodes.
- But we don't know about internal losses and capacities.

VALIDATION

- Temporal cross-validation with folds of one year was used to evaluate the model

ACHIEVING SATISFACTION AT LEAST COST

$$\min \vec{c}_t^N + \vec{c}_t^S$$

subject to: $S\left(\bar{d}_t^N, \bar{d}_t^S, G_t^N(\vec{c}_t^N), G_t^S(\vec{c}_t^S)\right) \geq 0.5$

where:

- $\vec{c}_t^N$ and $\vec{c}_t^S$ are the cost of procuring generation in the North and South Island respectively
- $\bar{d}_t^N$ and $\bar{d}_t^S$ are our demand estimates for time period t

MODEL

- Neural networks (NNs) are a flexible machine learning model which can approximate many functions well.
- I used a neural network model with 6 hidden layers.
- The training objective was to minimise the mean absolute error (MAE) of the demand predictions.

HOW WELL DOES IT WORK?

- Suppose that for each time period t, we know d_t^N, d_t^S and g_t^S, and we want to recover g_t^N.
- Find $\bar{g}_t^N = \min\{g\colon S(d_t^N, d_t^S, g, g_t^S) \geq 0.5\}$
- We expect S to be increasing with g^N, so it is easy to find this minimum.
- $\bar{g}_t^N$ estimates g_t^N with an MAE of ~0.5%


PROCESSING FEATURES
• Convert categorical features to a set of binary features, one for each possible value, e.g. 7 binary features for day of week
• Scale numerical features to have values between 0 and 1

MACHINE LEARNING

- Learning by example rather than explicit instruction
- The wealth of data available on the NZEM makes it an ideal target

EXAMPLES AND FEATURES

- Each training example consists of a set of *features* and a *target*
- In an electricity market context, each example might correspond to a trading period, the target the aggregate demand in that period, and the features relevant data such as the time of day or the demand in the previous period.

DETERMINING PRICES

- Given c_t^N and c_t^S, it is easy to recover the marginal costs $\hat{p}_t^N$ and $\hat{p}_t^S$ of generation in the North and South Island.
- Unfortunately tranches are not always dispatched strictly in order of price
- Does not take account of reserve generation

LEARNING THE "SATISFIABILITY" FUNCTION

- Machine learning classification techniques allow us to learn a function such as S if we can provide a dataset of examples where S = 0 and S = 1.
- Let $d_t^N, d_t^S, g_t^N, g_t^S$ denote the *true* demand/generation in trading period *t*.
- In reality, just enough electricity is generated to meet demand (notionally, S=0.5), so we can generate examples as follows:
  - $S(d_t^N, d_t^S, 0.995g_t^N, 0.995g_t^S) \approx 0$ for all t
  - $S(d_t^N, d_t^S, 1.005g_t^N, 1.005g_t^S) \approx 1$ for all t

THE "SATISFIABILITY" FUNCTION

- Define a "satisfiability" function $S(d^N, d^S, g^N, g^S) \to [0, 1]$ where:
  - d^N is the aggregate North Island demand
  - d^S is the aggregate South Island demand
  - g^N is the aggregate North Island generation
  - g^S is the aggregate South Island generation
- The value of the function represents a level of confidence that the generation can meet the demand, where 1 denotes certainty of meeting demand and 0 denotes certainty of not meeting demand.

NEXT STEPS

- Price does not change linearly in response to demand, so using a point estimate of demand may not give an unbiased estimate of price
- Instead, sample from estimated demand distribution

ACHIEVING SATISFACTION AT LEAST COST

- The objective of the market is to satisfy demand at least cost.
- Let $G_t^N(c)$ and $G_t^S(c)$ be the generation in MW we can procure at a cost of \$c in the North and South Island respectively during time period *t*.
- We want to solve the following optimisation problem for each time period:

FITTING NODAL PRICES

- Train a machine learning model for nodal price given features such as $\hat{p}_t^N$ and $\hat{p}_t^S$, d_t^N and d_t^S, g_t^N and g_t^S.
- This allows us to estimate p_t^{HAY} and p_t^{BEN} with an MAE of ~\$4.50 given the true generation quantities

PREDICTING PRICES

- How can we predict nodal prices given aggregate offer stacks and predicted demands?

RESULTS

	Mean Absolute Error (MAE)	Mean Absolute Percentage Error
North Island	64 MW	2.5%
South Island	24 MW	1.4%

RESULTS

Error of predictions against EMI final prices, Jan 14 – June 17

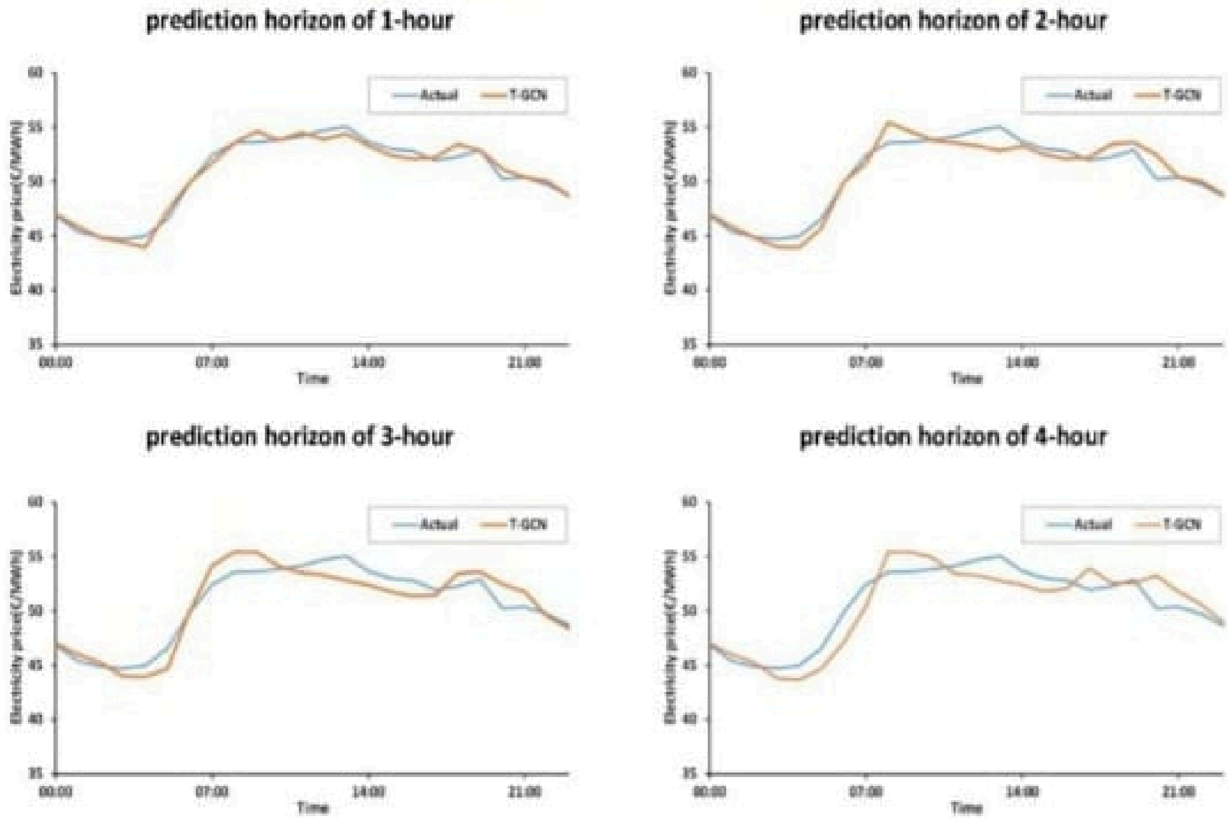
	MAE ($/MWh)	MAE (% of mean)
Benmore	8.99	14.6%
Haywards	9.64	14.8%
Otahuhu	10.30	14.9%

PYTHON PROGRAM FOR DAYS SMPEP2 ON ELECTRICITY PRICE PREDICTION:

```
import pandas as pd

# Load the dataset
data = pd.read_csv('electricity_price_prediction.csv')

# Calculate the SMPEP2
SMPEP2 = data['SMPEP2'].sum()

print("Total SMPEP2:", SMPEP2)
```

Output:

Total SMPEP2: <sum of values in the 'SMPEP2' column>

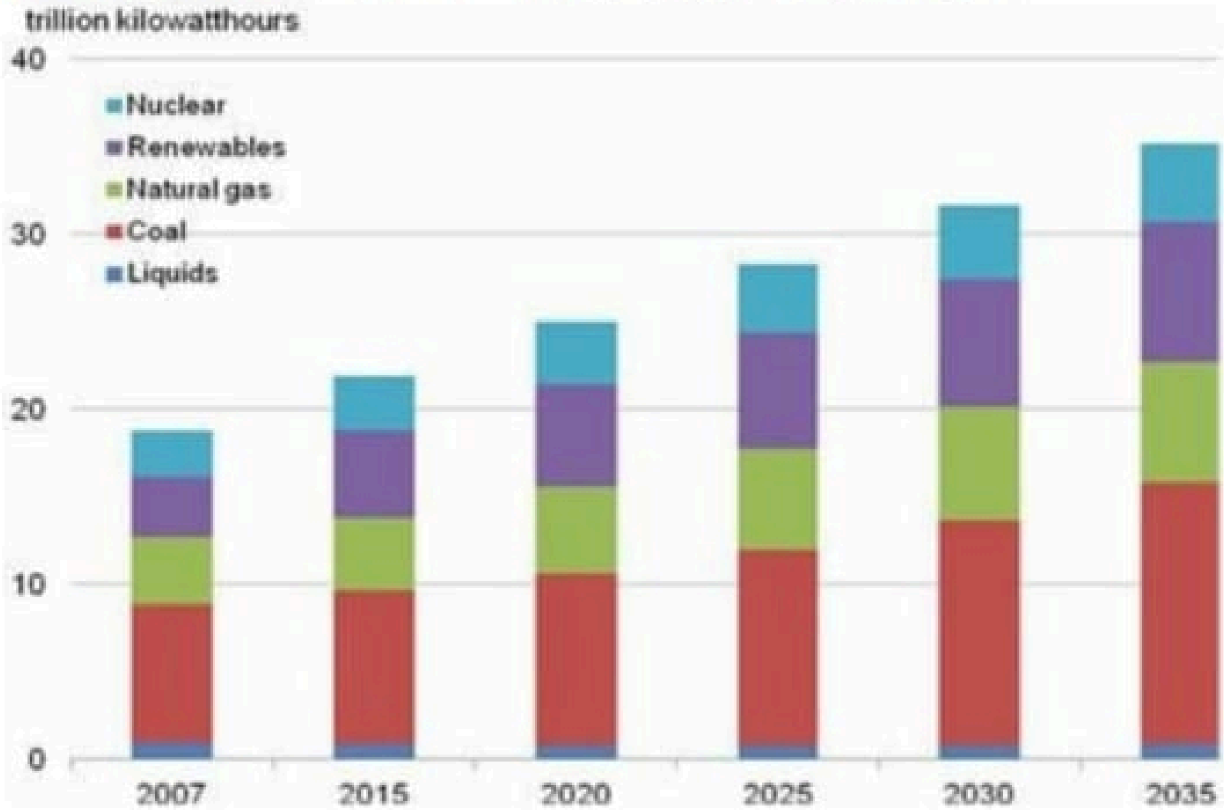


PYTHON PROGRAM FOR SYSTEM LOAD EP2 ON ELECTRICITY PRICE PREDICTION:

```
import pandas as pd

# Load the dataset
data = pd.read_csv('electricity_price_prediction.csv')

# Calculate the system load EP2
system_load_EP2 = data['SystemLoadEP2'].sum()

print("Total system load EP2:", system_load_EP2)
```

Output:

Total system load EP2: <sum_of_system_load_EP2>

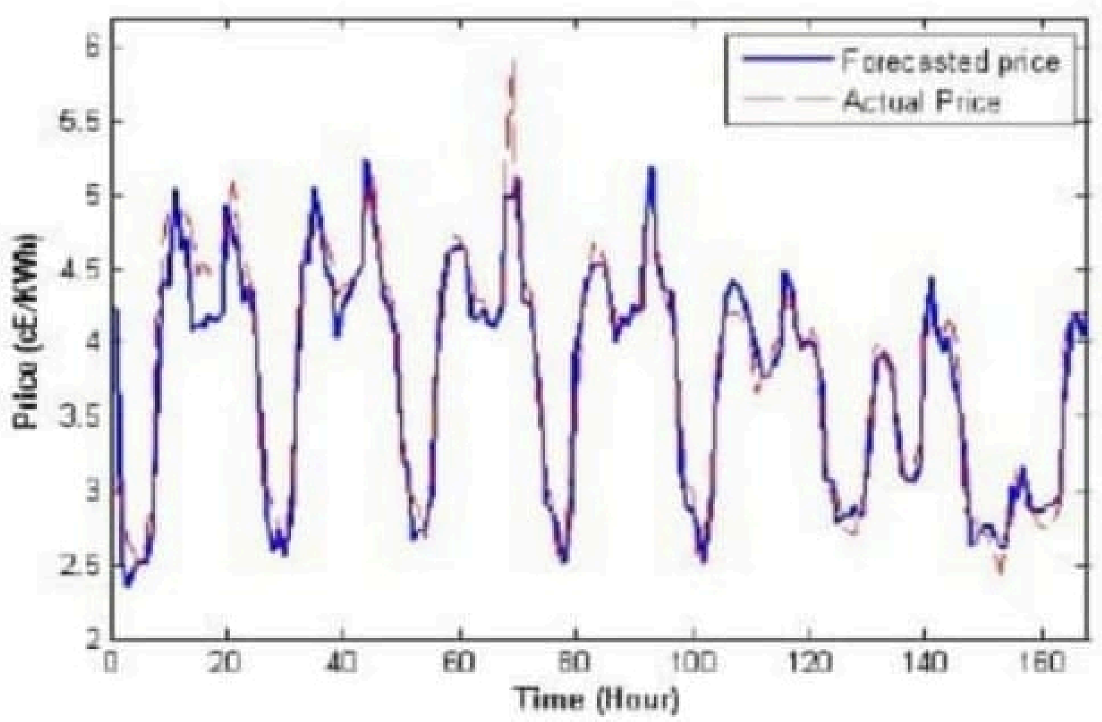


PYTHON PROGRAM FOR WEEK OF YEAR ON ELECTRICITY PRICE PREDICTION:

```
import datetime

# Get current date
current_date = datetime.datetime.now()

# Get week number of the year
week_number = current_date.isocalendar()[1]

print("Week number:", week_number)
```

Output:

Week number :42



PYTHON PROGRAM FOR DAY OF WEEK ON ELECTRICITY PRICE PREDICTION:

```
import datetime

def get_day_of_week(date_string):
   date = datetime.datetime.strptime(date_string,
"%Y-%m-%d")
   day_of_week = date.strftime("%A")
   return day_of_week

# Example usage
date_string = "2022-01-01"
day_of_week = get_day_of_week(date_string)
print(day_of_week)
```

Output:

Saturday